

RADC-TR-71-175  
Technical Report  
June 1971



AD737131

NETWORK INFORMATION CENTER AND  
COMPUTER AUGMENTED TEAM INTERACTION

Augmentation Research Center  
Stanford Research Institute

Sponsored by  
Advanced Research Projects Agency  
ARPA Order No. 967

Approved for public release;  
distribution unlimited.

The views and conclusions contained in this document are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of the Advanced Research Projects Agency or the U.S. Government.

Reproduced by  
NATIONAL TECHNICAL  
INFORMATION SERVICE  
Springfield, Va. 22151

Rome Air Development Center  
Air Force Systems Command  
Griffiss Air Force Base, New York

D D C  
REF ID: A737131  
FEB 18 1972  
RECORDED  
B

DISTRIBUTION STATEMENT A

Approved for public release

When US Government drawings, specifications, or other data are used for any purpose other than a definitely related government procurement operation, the government thereby incurs no responsibility nor any obligation whatsoever; and the fact that the government may have formulated, furnished, or in any way supplied the said drawings, specifications, or other data is not to be regarded, by implication or otherwise, as in any manner licensing the holder or any other person or corporation, or conveying any rights or permission to manufacture, use, or sell any patented invention that may in any way be related thereto.

Approved for	
CFSTI	WHITE SECTION <input checked="" type="checkbox"/>
DOC	COFF SECTION <input type="checkbox"/>
UNANNOUNCED	<input type="checkbox"/>
JUSTIFICATION	.....
BY	.....
DISTRIBUTION/AVAILABILITY CODES	
DIST.	AVAIL. AND/OR SPECIAL
A	

Do not return this copy. Retain or destroy.

## UNCLASSIFIED

## Security Classification

## DOCUMENT CONTROL DATA - R &amp; D

(Security classification of title, body of abstract and indexing annotation must be entered when the overall report is classified)

1. ORIGINATING ACTIVITY (Corporate author) Stanford Research Institute 333 Ravenswood Avenue Menlo Park, California 94025		2a. REPORT SECURITY CLASSIFICATION <b>UNCLASSIFIED</b>
3. REPORT TITLE "Network Information Center and Computer Augmented Team Interaction"		
4. DESCRIPTIVE NOTES (Type of report and inclusive dates) Interim Technical Report 8 Feb 70 - 8 Feb 71		
5. AUTHOR(S) (First name, middle initial, last name)  Douglas C. Engelbart		
6. REPORT DATE 30 Jun 71	7a. TOTAL NO. OF PAGES <i>104 105</i>	7b. NO. OF REFS 21
8. CONTRACT OR GRANT NO F30602-70-C-0219	9a. ORIGINATOR'S REPORT NUMBER(S) <b>SRI Project 8457</b>	
9. PROJECT NO. 0967	9b. OTHER REPORT NO(S) (Any other numbers that may be assigned this report)  <b>RADC-TR-71-175</b>	
10. DISTRIBUTION STATEMENT  Approved for public release; distribution unlimited		
11. SUPPLEMENTARY NOTES RADC (ISIM) D. Stone GAFB, NY 13440	Monitored by	12. SPONSORING MILITARY ACTIVITY Advanced Research Projects Agency 1400 Wilson Blvd. Arlington VA 22209

## 13. ABSTRACT

During 1970 SRI's Augmentation Research Center took part in preliminary operation of the ARPA network, made several important improvements in the ARC operating system's efficiency and features for users, and began installation of a new computer.

Conversion from an XDS 940 to a DEC PDP-10, which was in process in February 1971, has delayed full operation on the ARPA network.

However, the network has been used both in software development and in trial runs of the Network Information Center. Initial software for the Network Information Center was completed and documents have been rapidly accumulating. Other new hardware includes Univac drums and various remote terminals. New software includes redesign of the core of our NLS, development of higher level processes such as executable text, and ready use of content analysers in automated clerical procedures. New features for users include, among other things, an on-line Journal comparable both to a daily periodical and to archival journals, and calculator.

14. KEY WORDS	LINK A		LINK B		LINK C	
	ROLE	WT	ROLE	WT	ROLE	WT
Interactive Computing System Computer Network						

**BLANK PAGE**

NETWORK INFORMATION CENTER AND  
COMPUTER AUGMENTED TEAM INTERACTION

Contractor: Stanford Research Institute  
Contract Number: F30602-70-C-0219  
Effective Date of Contract: 8 February 1970  
Contract Expiration Date: 8 February 1972  
Amount of Contract: \$2,410,480.00  
Program Code Number: 62706D

Principal Investigator: D. C. Engelbart  
Phone: 415 326-6200

Project Engineer: D. L. Stone  
Phone: 315 330-3857

Approved for public release;  
distribution unlimited.

This research was supported by the  
Advanced Research Projects Agency  
of the Department of Defense and  
was monitored by D. L. Stone RADC  
(ISIM), GAFB, NY 13440 under Con-  
tract F30602-70-C-0219.

## PUBLICATION REVIEW

This technical report has been reviewed and is approved.

"Network Information Center and Computer Augmented Team Interaction"

Duane Stone  
RADC Project Engineer

ABSTRACT

1

During 1970 SRI's Augmentation Research Center took part in preliminary operation of the ARPA network, made several important improvements in the ARC operating system's efficiency and features for users, and began installation of a new computer.

1a

Conversion from an XDS 940 to a DEC PDP-10, which was in process in February 1971, has delayed full operation on the ARPA network.

1a1

However, the network has been used both in software development and in trial runs of the Network Information Center. Initial software for the Network Information Center was completed and documents have been rapidly accumulating. Other new hardware includes UNIVAC drums and various remote terminals. New software includes redesign of the core of our NLS, development of higher level processes such as executable text, and ready use of content analysers in automated clerical procedures. New features for users include, among other things, an online Journal comparable both to a daily periodical and to archival journals, and a calculator.

1a2

CREDIT

2

The research reported here is the product of conceptual, design, and development work by a large number of persons; the program has been active as a coordinated team effort since 1965.

2a

1970's work involved the whole ARC staff:

2b

Walter L Bass, Roger D Bates,  
Vernon R Baughman, Mary S Church,  
William S Duvall, Douglas C Englebart  
Martin E Hardy, J David Hopper,  
Charles H Irby, Milared E Jernigan,  
Harvey G Lehtman, John T Melvin,  
Jeffrey C Peters, Jeanne B North,  
James C Norton, Dirk H van Nouhuys,  
Cynthia Page, Bruce L Parsley,  
William H Paxton, Jake Ratliff,  
Barbara E Now, Edwin K Van De Kiet,  
and Kenneth E Victor.

2b1

in addition two consultants:

2c

Don I Andrews and James A Fadiman,

2c1

and the following former members of the staff:

2d

Geoffrey H Ball, Frederick van den Bosch,  
Mary G Caluweli, Roberta A Carillon,  
David G Casseres, Ann K Geoffrion,  
Jared H Harris, William K English,  
Martha E Trundy, and John M Yarborough.

2d1

CONTENTS

STATEMENT NUMBER

I ABSTRACT.....	1
II CREDIT.....	2
III CONTENTS.....	3
IV SUMMARY.....	4
A Introduction.....	4a
B Highlights of the Contract Year.....	4b
C Plans for 1971.....	4c
V ARPA NETWORK PARTICIPATION.....	5
A The Network Information Center.....	5a
B Connection to the ARPA Network.....	5b
VI CHANGING FROM AN XDS-940 to a PDP-10.....	6
A Hardware Transfer to the PDP-10.....	6a
B Compiler Transfer.....	6b
C NLS/TODAS Transfer.....	6c
VII NEW FEATURES IN 1970.....	7
A New Tools for Users.....	7a
B Core NLS.....	7b
C New Hardware Tools.....	7c
D Higher Level Processes.....	7d
E Design Team Augmentation.....	7e
VIII PLANS FOR 1971.....	8

IX GLOSSARY.....	9
X REFERENCES..AND.PUBLICATIONS.....	10
XI APPENDICES.....	11
A I/O .....	11a
B UNIVAC Drum System.....	11b
C Bryant Disc System.....	11c

IV. SUMMARY 4

A. INTRODUCTION 4a

1. This report covers the year 1970 of research in a continuous program at the Augmentation Research Center of the Information Science Laboratory of Stanford Research Institute, supported by ARPA under RADC contract F30602-70-C-0219. 4a1
2. The research reported is aimed at the development of online computer aids for augmenting the performance of individuals and teams engaged in intellectual work, and the development of the Network Information Center for the ARPA Computer Network. The report covers hardware and software development, applications in several areas, and a summary of plans for the continuation of the research during 1971. 4a2

B. HIGHLIGHTS OF THE CONTRACT YEAR 4b

1. During 1970 we devoted our attention especially to our continuing effort to improve the efficiency of our online system and polish and strengthen its usefulness to systems programming, to working with the ARPA Network, and to augmentation of distributed teams. During the latter part of the year we were deeply involved with translating our software into forms compatible with a PDP-10 and with choosing and connecting its peripheral equipment. 4b1
2. We have named an important new group of tools for users developed in 1970 "Higher Level Processes". They are routines in which the basic user features of our online system are building blocks in construction of programs that carry out specific, rather complicated tasks such as changing the order of a citation index and at the same time the format of the citations. Important Higher Level Processes are the rewritten Content Analyzer, the Analyzer Formatter, the Collector Sorter, and Executable Text. 4b2
3. We added an arithmetic and algebraic calculator package to our online system. 4b3
4. We have recently been working more with the goal of augmenting teams performing work that is distributed in time, space, and discipline. By way of communication and archival and managerial record keeping, we added a mail system and a Journal system. Any user may write a mail message from his terminal to any other users. The message is automatically brought to the recipient's attention when

Network Information Center  
and Computer Augmented Team Interaction

he logs in. Mail has been particularly useful to our people temporarily or permanently at a distance from the Center. Mail messages automatically become part of the Journal. The Journal is an online repository of the thoughts, records, baselines, and evolving designs of the group. In our community it serves the function that academic journals serve for communities defined by disciplines except that publication takes about a day. Recent entries in the journal are held on disc and magnetic tape, older entries on paper and magnetic tape. Online is an index to the complete journal, including various retrieving aids such as sorting by title words.

4b4

5. Our participation in the ARPA Network included: using University of Utah's PDP-10 via the Network to aid in our transfer to a new PDP-10, and development of the Network Information Center (NIC). In using the Net to re-program our PDP-10 we typically sent blocks to UTAH that consisted of relocatable binary data produced by compilers executing in our XDS 940 and producing code for the 10. The data was stored on a disc at Utah by the network control program so that someone here could reconnect and call on the Utah loader for the transmitted file. We found this service so useful that we added multiplexing at this end so that three of our programmers could use the Utah system at once. The link to Utah operated daily from August 1970 through January 1971 and constituted the most substantial data transmission over the Net to that date.

4b5

6. At ARC we established a collection of documents that form the basis of the Network Information Center, established online techniques for handling the documents, and most important, began working dialog with the other centers. The combination of our reference data storage techniques with our programming allows retrieving documents according to a variety of attributes and combinations thereof. E.g., year of publication combined with author, or sponsoring institution. We organized with the other sites on the Network to establish Station Agents to handle their interaction with the Network Information Center and supplied the Station Agents with a catalog of their collection and other working materials. To stimulate dialog, pending full operation by connected computers, we set up a central telephone exchange and a system for circulating documents and memos by U.S. Mail through the NIC, including an intra-Net document numbering system.

4b6

7. In the Spring of 1970 we decided that DEC's PDP-10 with associated software and paging box from BBN might be a way

Network Information Center  
and Computer Augmented Team Interaction

to increase the number of consoles and displays available to us, to strengthen our system in other ways, and to insure a system that could be expanded further with ease. In June after investigating several competing machines, we ordered a PDP-10 which was delivered in September. Our 940 was removed February 1, 1971. Associated equipment for the PDP-10 includes 128K of 1.0-microsecond core and the BB&N Paging Box. After studying the various alternatives, we retained from the 940 system a 32K-word Ampex external core, UNIVAC drums as a swapping device, and a Bryant Disc for mass storage. A drum/disc interface, an interface for the external core system, and an I/O control box were built locally to our specifications.

4b7

8. Re-programming for the 10 has created the necessity and opportunity for thorough-going revision of our software. Our online system which had been written in a special language, SPL, has been rewritten in L10, a language much more machine independent and more flexible in application. Our NLS has been rationalized to allow more routines to call on other routines. Display routines have been changed to allow division into up to eight areas which the user can load and edit independently. Many other features such as Mail, Journal, calculator have been substantially improved in the transfer.

4b8

C. PLANS FOR 1971

4c

1. Early in 1971 we will complete transfer to the PDP-10, develop further and operate the Network Information Center, give more powerful tools to the users of our Dialog Support System, expand our Baseline Planning system, and make a variety of specific operational improvements.

4c1

2. we have established a three-step schedule of increasing interactive support to the other members of the AKPA network. Uncertainties in the capacity of our new computer system when it is finally tuned, and in the load interactive service over the net will place on us make it difficult to estimate the number of users we will support at exact dates. We are proceeding with the following general plan:

4c2

In stage 0, beginning in mid June, we will offer experimental access to the NIC to a limited number of West Coast sites and to RADC and will offer a two-day course at SRI in our typewriter terminal online language which has been rewritten to provide users with more of the power of NLS.

4c2a

Network Information Center  
and Computer Augmented Team Interaction

SRI-ARC 1 JULY 71 0277  
Summary

In stage 1, beginning in mid August, we will offer further instruction, and an operational access to NLS to 4-8 simultaneous users. 4c2b

In stage 2, beginning in October, we will offer message delivery online to remote sites, a deferred execution system for offline preparation of files for NLS use, and access to more users. 4c2c

3. We will improve our Dialog Support System by further automation of entry of items into the Journal and of study of the documents in the Journal. 4c3

A command language in which the Journal may interrogate the user for information necessary to identification and automated retrieval of the document will make entry simpler and more effective. 4c3a

Devices such as automatic construction of links between documents and generation of sets of documents, along with set manipulation commands, will facilitate study. 4c3b

4. We will manage our daily activities more and more by means of our Baseline Planning system and develop new subsystems within it to, for example, filter out and record various useful views of the organized planning of the whole group. 4c4

5. Modular programming will make it much easier to transfer all or part of NLS to users outside of ARC. Design and implementation of a preliminary system will take place in 1971. 4c5

6. We have detailed plans for a variety of improvements in our NLS and Executive, most of which directly support NIC operations. 4c6

Network Information Center  
and Computer Augmented Team Interaction

V. ARPA NETWORK PARTICIPATION	5
A. THE NETWORK INFORMATION CENTER.	5a
1. Goals, strategy, and philosophy	5a1
We see NIC's primary role in the Network experiment as support for Network participants: tools, techniques, and services of computer, storage, and people. We aim eventually to provide highly interactive information services that will be valuable to a dynamic clientele and we hope to facilitate highly responsive dialog between and with them. But basically we can only serve as a supporting agent toward these ends. Emergence of a significant dialog will require active Network participation both among the nodes and with NIC as it learns how to serve their information needs.	5a1a
Some of the services we will provide are:	5a1b
Collection and storage of a wide range of Network-relevant reference information (in the NIC Master Collection).	5a1b1
Online service over this collection for querying, browsing, and retrieving -- designed to serve a range of terminals (typewriters to CRTs).	5a1b2
A coordinated set of offline reference materials (indexes, etc.).	5a1b3
A communication service in which there will be direct, interactive, and sophisticated handling of messages -- their composition, delivery, verification, storage, and retrieval (they become part of the master collection).	5a1b4
A natural means for linking messages to each other and to any other items in our collection, to produce an organically developing network of dialog items whose search, study, and integrative manipulation will be supported by our computer aids as a basic NIC service.	5a1b5
We have been heavily involved in preparing ourselves toward these ends, both in range of services and in capacity to handle customers. It is quite evident that these ends will be reached only by steady evolution, throughout the range of site facilities,	

Network Information Center  
and Computer Augmented Team Interaction

supporting technologies, user methods, and user  
skills.

5a06

2. Starting up

5a2

For some time, we were concerned with the question: how could we launch a new, experimental service for a clientele that we didn't see or hear, where the service was to be designed for a degree of computerized communication that hadn't yet emerged, but where it was disturbingly apparent that the proper performance of our declared function could significantly accelerate that emergence? What was needed badly in order that we at NIC could produce some service, and also, we thought, in order that the Network could become alive was for a sizable amount of stimulating and visible dialog to take place. To this end, we decided to dedicate most of our NIC service energy over the past several months toward stimulating and supporting such visible dialog -- which is the reason for the "Network Dialog System" development.

5a2a

To provide a useful initial service to the Network Community, and also to give our evolutionary process a starting place, we adopted the initial-stage design described below for visible dialog.

5a2b

3. Dialog

5a3

By "visible dialog" we mean messages and memos that become a public record available to all potential Network participants, for later reference, citation, retrieval, or browsing; where people other than those involved in a given exchange are welcomed -- and helped -- to discover its existence and contribute questions and additions that in turn are incorporated as part of the recorded dialog. At first the media are whatever communication means can be best used at the moment -- mail, telephone, and the ARPA Network as it becomes more and more functional.

5a3a

To encourage communication initially, we are maintaining a NIC telephone system that provides toll-free service, especially for Station Agents and Liaison people although open to others. The system makes use of a commercial, after-hours answering service, and is responsive to special needs around the clock.

5a3b

4. Station Agent and Liaison

5a4

Network Information Center  
and Computer Augmented Team Interaction

The Network Dialog System involves at least two specially assigned people at each site, who are as soon as possible to be provided with an online typewriter at a specially designed Reference and Communication Station. Besides the typewriter, the Station is expected to include a telephone and certain hard-copy reference materials supplied by NIC.

5a4a

The Station Agent helps the NIC with local services performed in our behalf, such as seeing that messages are delivered to local people, helping people learn how to use NIC services, updating locally held hard-copy, NIC reference material (according to instructions and materials supplied by us), helping local users find needed information among the various hard-copy materials that will comprise an important part of our early services to each locale; and providing feedback to us about needs and possibilities for improving our services.

5a4b

Each site also has a Liaison Contact available to the Agent for technical backup. He is usually a technically oriented person who is used to learning online techniques, who understands at least enough of the Network technology to interpret technical questions accurately and to pursue their answers intelligently. He is also expected to field technically oriented questions and requests from other Network sites.

5a4c

In particular, the station Agent will need a certain amount of consistent, supportive help in learning about technical details associated with some of these tasks -- we need each Liaison Contact to provide this (thus helping to form a working team, with whom we at NIC can work consistently, and about whom people at the site can feel comfortable in handling the reference and communication aspect of their total "Network interface"). Both people are becoming useful sources of Network folklore for people at the sites.

5a4d

## 5. Manipulation

5a5

We are now set up to handle the transmission/distribution of such material as submitted to us, and to provide storage, indexing, retrieval and access to the accumulated material -- in hard copy mailed media and/or by online access, whatever it takes to get things rolling. One-sentence messages, very informal memos, tentative plans, "CQ calls" seeking

support or interaction, announcements of up-down-changes etc., arguments about how things should be done -- telephoned to us, mailed in long-hand or typescript, composed via Network access to our online editing system, sent or transmitted as a file composed on your editing system -- we are trying to handle them all. 5a5a

After our transfer to the PDP-10 is completed, we will be ready to provide online interaction, in typewriter mode, for initial experimentation, for editing, for access to dialog material, etc. (holding off on more general access to reference material for the time being). Local Station Agents will be supplied with the reference information necessary to link to us and we will offer first to check them out and the Liaison men. They will (we hope) then check out other users. 5a5b

6. NIC Station Collection 5a6

Collection: 5a6a

Physically, we have over 5,000 items, mainly external documents, in ARC's Master Collection. The NIC Collection is a subset of the Master Collection. We estimate that 500 to 800 of the items currently held will eventually prove relevant to the NIC clientele. At present some 300 of our most relevant documents have been replicated and a set installed at each Station together with a computer-generated, hard copy shelf listing and index by number, author, and titleword. 5a6a1

We have isolated several hundred items that seem relevant now. These will be included soon in the NIC "Subcollection". 5a6a2

We are providing for steady addition from messages, memos, survey summaries, formal Network documentation, etc. 5a6a3

The most significant documents to the NIC Collection from volume and content relevance standpoints are those currently being added through dialog between network people and through collection and publishing of information describing network facilities, interests, and resources. 5a6a4

Catalogs and indexes 5a6b

We have developed common conventions for catalog entries over the entire ARC (and therefore NIC) collection. Online entry formats are being converted from old formats to consistent more easily searched formats. All current NIC and ARC Journal collections are in the new catalog formats. All new entries take the new form. Each entry is stored now as one long string, within an NLS statement, with special strings of characters that separate and identify the different data elements.

5a601

B. CONNECTION TO THE ARPA NETWORK.

5b

1. Our first hardware and software connection to the ARPA Network was completed in November of 1969 and is discussed in some detail in the references (ref.1). At the end of 1970 the hardware interface was still as described in that report and has been operating since that time with no difficulties.

5b01

2. The early software was definitely experimental. A preliminary network operating system was written which ran as a user program and allowed the login of one remote user over the Network and the simultaneous use of a remote facility by one local user.

5b02

3. Following this experimental system, work continued on a first-stage Network protocol operating at the monitor level.

5b03

Since there was no official Network protocol at that time, it was necessary to develop compatible protocol at at least two sites for debugging and experimental use of the facility. The University of Utah was chosen as the site for this activity, mainly because they were eager to cooperate in the experiment and personnel were available at ARC who were familiar with both the 940 operating system and the University of Utah's system.

5b3a

Programs at both sites were written primarily by ARC personnel. (They included the monitor level coding required to operate the hardware and to allow programs logical access both to the Network and user level programs.)

5b3b

The system when completed allowed a user at SRI to connect his teletype to Utah's time-sharing system with all the privileges and capabilities of a user locally connected at the University. Capabilities were also

provided for Utah use of the ARC facilities, although this feature was never thoroughly checked out. 5b3c

4. When we determined to convert to a PDP-10 we decided to use the PDP-10 at Utah to aid the software conversion effort. This new application required some modification of the temporary UTAH/ARC protocol at both ends. 5b4

Programs were modified to allow transmission of files so that blocks of data could be sent to Utah. Typically we sent blocks to Utah that consisted of relocatable binary data produced by compilers executing on the 940 and producing code for the PDP-10. Data transmitted from ARC was stored on a disc at Utah by the Network control program so that the sender at ARC could subsequently invoke his Network teletype connection and call the Utah loader to load the transmitted files. This arrangement gave the ARC programmer an extended debugging tool close at hand. We found this service so useful that multiplexing was added to both ends of the connection allowing three ARC users to work simultaneously with the Utah system. The link with Utah was in use daily from August 1970 to January 1971 for the modification and debugging of our NLS required to convert over to the PDP-10. 5b4a

5. ARC personnel have participated in the Network Working Group, and we have followed closely the development of the official Network protocol. Implementation of this protocol in the 940 was planned in detail, but the anticipated transfer to the PDP-10 and the lack of other operating protocols on the Network, obviated the 940 version. 5b5

VI. CHANGING FROM AN XDS-940 TO A PDP-10

6

A. HARDWARE TRANSFER TO THE PDP-10.

6a

1. As the Augmentation Research Center has evolved more and more to an online community, the needs for computer service have steadily increased. Early in 1969 when experience showed that the 940 would support only about 6 display consoles, studies were undertaken on various approaches to increasing the service capacity. At that time the 940 still offered the only time-sharing system suitable for our needs and within our budget. The most reasonable approach to getting more service seemed to be the use of a small computer subsystem in conjunction with the 940. Work on this approach continued and in January of 1970 a small computer was selected for the development of an experimental front-end subsystem.

6a1

While the small computer approach was being pursued, we were also keeping up with developments in other computers and time-sharing software. In the Spring of 1970 it became apparent that the PDP-10 with the TENEX software system and associated paging hardware being developed by BB&N would be a major contribution to the field of research time-sharing.

6a1a

When the PDP-10 became a real possibility we undertook a brief study of other available machines and associated time-sharing software. We considered in particular the XDS Sigma 7, the CDC 3300, and the Standard Computer IC-9000.

6a1a1

The last machine named would have been microprogrammed initially to emulate the 940 with an immediate increase in capacity due to the faster machine. Operations would have later been developed to more closely suit the needs of the ARC software system.

6a1a1a

Of these machines the Sigma 7 and the CDC 3300 were quickly eliminated by lack of available time-sharing software. We seriously considered the IC-9000, but its uncertain development schedule and the unpredictable effort required for further development of the microcode ruled it out.

6a1a1b

Investigation of price on the PDP-10 system, both from DEC and other sources, showed that it would be possible to replace the 940 with a significantly

larger PDP-10 system with a monthly lease slightly cheaper than that for the 940. Furthermore, the PDP-10 system would be much more open ended than the 940 system. Core memory could be expanded greatly, particularly through the BB&N Paging Box. A wide range of peripherals is available and additional processors can be added. This expandability, together with the immediate increase in service capacity and slightly lower cost, seemed to justify the expense of converting software to the new system. An order was placed for the PDP-10 facility in June 1970 and the system was delivered in late September. 6a2

2. Figure 1 is a block diagram of the PDP-10 facility. It consists of the following major units: 6a2

The equipment leased from DEC includes the KA10 processor, 8 banks of MA10 memory (for a total of 128K), two mag tape units and controller, two DEC tape units and controller, and a teletype scanner for 24 teletype lines. 6a2a

The BB&N pager connects between the KA10 processor and the memories. In conjunction with a set of hardware modifications to the KA10, the pager changes the core memory mapping mechanism. 6a2b

The UNIVAC drum system consists of four UNIVAC FH432 drums and a UNIVAC FH432/1782 controller. This system was our swapping medium on the XDS-940. It has a storage capacity of approximately 1 million 36-bit words. The drums turn at 7,200 rpm, with 2048 words per track, providing a transfer rate of 240,000 words per second and an average access time (for each drum) of 4 milliseconds. 6a2c

The Bryant disc system was also in use on the XDS 940. It consists of a Bryant Model 4061-A24-16 Disc with 13 data surfaces and a Bryant controller. 6a2d

The 24-bit Ampex Memory was in use on the 940 as an external memory system for display refreshing, network buffers, and line printer buffers. It was transferred to the PDP-10 system as an extra bank of directly addressable memory. 6a2e

Other equipment shown in the facility was previously in operation with the 940 and was already connected to the 24-bit External Core Memory (XCORE). It consists of the

following units that are described in more detail in the references (Ref.2.3): Two display systems for a total of 12 display consoles; Input device controller for input from the 12 consoles; Line printer controller; Network interface; Interface for a high-speed modem to drive the IMLAC display. 6a2f

3. The choices of equipment to make up the PDP-10 facility were governed primarily by the need to comply with the BB&N system to make maximum use of the TENEX software and our desire to minimize the cost of transferring to the new facility by employing the existing equipment wherever possible. 6a3

Since the decision to transfer to a PDP-10 was based on the development of the TENEX time-sharing system, paging hardware was essential to the system. BB&N developed a paging box and associated modifications to the PDP-10 processor in conjunction with the TENEX development and was the only reasonable source for such hardware. 6a3a

The amount of core ordered with the DEC system was determined by funds available for monthly lease and turned out to be 12dk of 1.0-microsecond core. 6a3b

For a swapping device, the obvious possibilities were the Bryant drum as used at BB&N, the UNIVAC drums already in use on the 940, and the swapping disc offered by DEC. 6a3c

We ruled out the DEC disc because of its slower transfer rate, but gave considerable study to the Bryant drum and the UNIVAC drums. Speed was the major focus of the study. 6a3cl

The Bryant drum rotates at 1800 rpm and can transfer up to 18 512-word pages in 34 milliseconds (one revolution). The UNIVAC drums, on the other hand, rotate at 7200 rpm and each one can transfer 4 pages per revolution. But, since there are 4 drums running asynchronously, the average maximum transfer will be about 13 pages in 34 milliseconds. 6a3cla

These rates are maximum. The percentage of possible transfers which are actually used depends on the length of the drum queue and the distribution of requests. 6a3clb

Our studies showed that for about 20 items in the queue with a uniform distribution over pages of the drum, the Bryant is able to use about two thirds of its possible transfer rate. The UNIVAC is able to give a higher actual transfer rate than the Bryant for queue lengths less than 20 because of the faster rotation and resulting lower latency.

6a3clc

In favor of the Bryant were lower cost and less software development because this drum is used by BB&N in their TENEX facility. In addition, changeover would be easier since the UNIVAC drums could be left on the 940 while getting the Bryant going on the PDP-10.

6a3cld

The UNIVAC drums appeared more reliable. There have been some bad experiences with head crashes on the Bryant drum, and with a single drum in the machine and few machines in the field a crash could mean being down for several months. (UNIVAC has many of these drums in the field, would be able to replace a bad unit in very short order, and the system could operate on three drums in the meantime.)

6a3c1e

Reliability and speed, as well as a somewhat indefinite delivery schedule on the Bryant drum, led us to the decision to use the UNIVAC drums with the PDP-10 system.

6a3c2

In the case of mass storage medium, our possibilities were the existing Bryant disc, or the addition of disc packs, such as the DEC KPO2 disc drives, or the IBM 2314. Here investigation showed that the Bryant disc had been designed for easy modification to 36-bit mode, and that interface cost would not be too high. Since we already owned the Bryant disc, it was significantly cheaper to use it than to add any other storage medium.

6a3d

#### 4. Adapting the Special Equipment

6a4

Three interface units were required to connect the non-DEC equipment to the PDP-10. These were: (1) a drum-disc interface; (2) an interface for the 24-bit external core system; and (3) an I/O control box to convert the PDP-10 I/O commands to signals expected by the equipment that previously operated on the 940. The

functions of these units are described in detail from  
the programmer's viewpoint in the Appendices. 6a4a

All these of these interface units were built to our  
specifications by Cybernex, Inc. of Palo Alto,  
California. 6a4b

In the construction of these units, DEC logic cards  
were used in some cases. In others cases it was  
cheaper to make up special boards using integrated  
circuit modules (dual-in-line packages). All panel  
indicators are light-emitting diodes driven directly  
from the logic circuits. All three units have fairly  
extensive control panels with indicators for data and  
major control signals plus switches for simulating  
data and control signals. These panels made  
debugging and maintenance much easier. 6a4b1

The Drum-Disc Interface: 6a4c

This unit connects the Bryant Disc Controller and the  
UNIVAC Drum Controller to a PDP-10 memory bus. Data  
rates for these units allow both to share a common  
memory bus. The drum has priority because its  
transfer rate is higher. Both devices may be  
transferring data simultaneously and memory bus  
multiplexing takes place cycle-by-cycle. 6a4c1

Control and interrupt signals for these units are  
processed through the I/O Control Multiplexor to  
avoid the necessity of connecting the I/O Bus in  
the drum-disc interface. 6a4c1a

The Bryant Disc Controller contains facilities for  
memory address and word-count registers and for  
interpreting command tables in core. Therefore,  
the portion of the interface handling this device  
simply transforms PDP-10 memory bus signals into a  
simulated XDS 940 memory connection. 6a4c1b

The UNIVAC drum portion of the interface, however,  
must provide word count and address registers and  
otherwise perform the functions of a UNIVAC 1108  
I/O channel, including the generation of function  
words to the drum system in response to signals  
from the I/O Control Multiplexor and the  
interpreting of status words from the drum system  
to generate status and interrupt signals. 6a4c1c

The External Core Interface:

6a4d

This unit connects the existing 24-bit core memory system to the PDP-10 processor memory bus. Viewed from the PDP-10, the External Core (XCORE) system performs exactly as if it were part of the PDP-10 main memory, with the exception of the missing 12 higher-order bits in each data word. These bits will be ignored when writing and will be supplied as zeroes when reading. Differences in memory cycle times are not significant because the PDP-10 memory is asynchronous.

6a4d1

The XCORE memory has no provision for a parity bit. The PDP-10 memory bus provides for this contingency through the ignore parity signal which is generated by the interface.

6a4d2

The XCORE bank was implemented on the 940 with an 8 port access switch designed to have exactly the same interface characteristics as the executive controller used on the 940 memory port (Ref.2). This access switch was modified to provide high priority for one port to connect to the PDP-10 and the XCORE interface unit was designed to convert PDP-10 memory bus signals to those required by the access switch interface.

6a4d3

Aside from coordinating the memory control signals on both sides, the principal function of the interface is to transform the negative logic pulse bus of the PDP-10 into the positive logic of the XCORE system.

6a4d4

I/O Control Box

6a4e

This unit processes I/O control signals for units connected to the 24-bit XCORE memory system and for the drum-disc interface. It generates command signals in response to instructions from the PDP-10, provides status bits that may be read by the PDP-10, and processes interrupts to the PDP-10 with interrupt mask and priority selection features.

6a4e1

5. Addition of the BB&N Paging Box

6a5

The Pager connects the processor to the memory. In conjunction with modifications to the processor it changes the core memory mapping mechanism so that core memory is allocated and protected in 512-word pages.

The address space is mapped for executive mode as well as User mode. The paging mechanism may be bypassed either by a direct reset switch or by a PDP-10 instruction to permit running of standard DEC software. 6a5a

To implement new instructions and to operate the Paging Box, fairly extensive modifications were required to the KALO processor. 6a5b

BB&N provided documentation for these modifications with the the Paging Box. This documentation was very complete. It included logic diagrams for all portions of the processor affected and complete wire lists for additions and deletions. These changes involving approximately 700 wires (576 additions and 148 deletions) required approximately four man-weeks of ARC personnel time and were successfully completed in two weeks elapsed time. 6a5b1

In the course of checking out these modifications, only two minor errors were found in the BB&N documentation, and the Paging Box functioned perfectly from the start, with no errors. This is highly commendable considering that ARC is the first customer for the TENEX-Paging Box system. 6a5b2

#### 6. Teletype Patching System 6a6

A teletype patching system was constructed by ARC personnel to provide flexible patching of teletype lines to various spots in the building, as well as to data sets. The patching facility includes local monitoring for maintenance and a variable character rate to accommodate a variety of terminals in use. 6a6a

Four character rates, 10, 15, 30 and 60 cps, can be increased to a total of eight selectable character rates. Speed for a local terminal is determined by appropriate jumpers in the connector on the terminal. Over the telephone, speed is selected by dialing a digit after connection to the computer via the data set. The speed of the dial-up connection may be changed at any time simply by pulling the telephone dial an appropriate number of times to step through the available speeds. 6a6a1

SKI-ARC 1 JULY 71 8277  
Changing from an XDS-940 to a PDP-10

B. COMPILER TRANSFER

6b

1. Introduction.

6b1

NLS on the 940 was written in a machine-dependent language called MOL. The MOL compiler was written in a compiler writing language called TREE META. MOL was a systems programming, high level language, specially written for the 940 and for writing NLS. The MOL (and also TREE META) were written to operate under NLS as a sub-system. TREE META is written in its own language, that is, it compiles itself. Parts of NLS were written in a language called SPL (special purpose language), which is also a META generated language.

6b1a

Changing from the 940 to the PDP-10 provided an opportunity to redesign NLS and other subsystems to a degree that the continuous press of other work did not allow us on the 940. The redesigning provided more flexibility and better service and made it possible to extend NLS much more than we could on the 940. No suitable programming language existed for the PDP-10 which could correspond to MOL. In addition, we had several ideas about combining SPL and MOL into one language.

6b1b

2. Approach.

6b2

The approach we decided on was basically to convert TREE META to run on the PDP-10, design a new systems programming language, L10, and compile it on the PDP-10 with TREE META. We decided that TREE META was powerful and useful enough to warrant transferring to the PDP-10. In addition, the PDP-10 is a much more suitable machine for TREE META. We also wished to make several additions and changes to TREE META itself, which we could not do on the 940.

6b2a

L10 was designed to take advantage of features that were available on the PDP-10 and not on the 940. The L10 language was specified in advance, and the NLS system was rewritten carefully (using the NLS system on the 940) starting about 8 months before it was actually compiled on the PDP-10.

6b2a1

3. Outline of the conversion.

6b3

The steps in converting TREE META to the PDP-10, and getting L10 running on the PDP-10, were as follows:

6b3a

First, it should be explained that TREE META is a program that compiles symbolic files which describe the syntax and code production rules for a language. The result is a binary file which is given to a loader. That binary file must be accompanied by a library of procedures -- which is common to all TREE-META-generated compilers. 6b3a1

we will use an upper case letter to represent a symbolic file, and a lower case letter to represent a binary file. Compilation will be written thus: 6b3a2

{p+l}(Y) -> y 6b3a2a

which reads: program p combined with program l (the library) compiles symbolic file Y to produce binary file y. 6b3a2b

The situation on the 940 was as follows: 6b3b

The current TREE META symbolics were called T2. The T2 compiler would compile MOL, the symbolics for the current MOL compiler. The TREE META library was written in MOL and was called L. The current running TREE META was thus {t2+l} and MOL was {mol+l}. Notice that: 6b3b1

{t2+l}(T2) -> t2 6b3b2

{t2+l}(MOL) -> mol 6b3b3

{mol+l}(L) -> l 6b3b4

The first step was to alter the library (L) to produce 36-bit binary files for the PDP-10 loader, rather than 24-bit binary files for the 940 DDT. We will call the new library L36: 6b3c

{mol+l}(L36) -> l36 6b3c1

We also altered T2 to produce 36-bit instructions, and to produce code to run with l36. The modified TREE META was called T2.5. 6b3d

{t2+l}(T2.5) -> t2.5 6b3d1

The next step was to write a compiler like L10, but one that would run on the 940 and produce binary loadable files for a PDP-10. It was written carefully to compile

SMI-ARC 1 JULY 71 0277  
Changing from an XDS-940 to a PDP-10

a subset of L10 (because the 940 memory was smaller than  
the PDP-10). We called it L940: 6b3e

(t2.5+1)(L940) -> l940 6b3el

The L940 compiler would compile L10 programs to load on  
a PDP-10, provided they used only the syntax included in  
L940. The library that would run with TREE META  
programs on the PDP-10 was then written in L940. We  
called it LIB: 6b3f

(l940+136)(LIB) -> lib 6b3fl

this library could be loaded on a PDP-10 6b3fla

At the same time the real TREE META for the PDP-10 was  
written in the T2.5 meta language and compiled on the  
940. Call it T3: 6b3g

(t2.5+1)(T3) -> t3 6b3gl

This T3 was then ready to run on the 940 and produce  
PDP-10 code. In particular: 6b3h

(t3+136)(T3) -> t310 6b3hl

which is ready to load on a PDP-10. 6b3hla

Also, L10 was written in the T3 language, including the  
full syntax this time, and using all of the new features  
in T3. It had to be compiled on a PDP-10 due to the  
restricted size of the 940: 6b3i

(t310+lib)(L10) -> l10 (compilation on a PDP-10) 6b3il

running L10 on a PDP-10 is represented by (l10+lib). 6b3j

Actually, it was somewhat more complicated than the  
description above because of these problems: 6b3k

Symbolic programs on the 940 are 8-bit non-ASCII  
characters. On the PDP-10, characters are 7-bit  
ASCII. It was easier to introduce one extra step of  
META compilers to convert the literal strings inside  
the binary files than it would have been to write  
code to translate 8-bit, 3-character-per-24-bit-word  
text streams to 7-bit, 5-character-per-36-bit-word  
text streams, on a 940. 6b3kl

Some features we wished to include in the new TREE  
META (T3) could not be reasonably compiled by the 940  
TREE META, and an extra step was made to get to T3. 6b3k2

4. Method of debugging: the Network 6b4

Arrangements were made to use the PDP-10 at the  
University of Utah for debugging our compilers. 6b4a

Programs were written on both ends to allow 940 users  
to send files to Utah's file system, and to log in  
and use the Utah system. 6b4al

Programs (primarily L36 and T3) were checked out by  
running them on the 940 and sending the binary results  
to UTAH and loading. Format errors and so on were found  
by checking the binary image or the results of the  
loading in Utah. 6b4b

When programs could be successfully loaded in Utah,  
symbolic test files were sent to Utah, and the compilers  
were tested (L10, T3). The results (PDP-10 binary  
loadable files) were loaded in Utah and checked. 6b4c

In any event, bugs were corrected in the symbolics on  
the 940, and the necessary compilations were done again  
and tested. And so on. 6b4d

This work was primarily done during off hours in order  
not to load our 940 too much, and in order to get  
reasonable response from Utah. 6b4e

The alternatives would have been to have both the 940  
and our PDP-10 available to users for several months,  
which would have been quite costly, or to use another  
PDP-10, which would have involved at best carrying  
magnetic tapes back and forth between computers. The  
conversion would have taken perhaps three times as long. 6b4f

The actual transfer to our PDP-10 was simple. Programs  
were written to transfer files through XCURE (which is  
part of the PDP-10 addressable memory). The PDP-10  
loadable binary files, and symbolic files were sent  
across to our PDP-10 and loaded. 6b4g

C. NLS/TODAS TRANSFER. 6c

1. The transfer to the PDP-10 demanded certain software  
changes in our NLS and offered a particularly good

opportunity to make others. Here we list them. For the approximate baseline from which we here depart, see reference 1.

6cl

Reorganization of NLS:

6cla

The online system (NLS) has been modified so that the user specifies his terminal device and NLS provides the appropriate command parser and character definitions for that device. This modification subsumes the 940 NLS/TODAS subsystems. NLS was also reorganized to allow the user access to the typeWriter-oriented and display-oriented command parsers for NLS and the parameter specification and executor for each command--this also make possible separation of NLS command specification from the (core-NLS) file manipulation, with perhaps a network in between.

6clal

New Capabilities:

6clb

File System

6clb1

The file system implemented in the PDP-10 NLS system functions as does that of the 940 NLS system, but allows more core space for file blocks, applies paging to those file blocks, and allows for more than one file.

6clb1a

In addition, the "working copy" of the 940 system has been replaced by a "partial copy" which contains only the blocks of the original file which have actually been changed by the user.

6clb1a1

Also, only one user may now modify a source file at one time. The partial copies are retained until the user writes the changed file onto a source file or explicitly deletes the partial copy.

6clb1a1a

As before we will have files, called "checkpoints", onto which copies of the partial copies are written for security and convenience. There will now be two checkpoints for each source file being modified. Those partial copy blocks which have changed since the second-to-the-last checkpoint are periodically copied to the oldest of two

checkpoint files, the frequency depending on  
the user's activity and a maximum amount of  
time between checkpoints.

6clbia2

Display Areas

6clb2

Unlike the 940 NLS system, the TENEX NLS system  
allows the user to subdivide the text area of his  
screen into rectangular, non-overlapping display  
areas. We provide the user with commands to split  
extant display areas into two display areas, move  
the boundaries of display areas, and erase the  
display image from a single area. The user may  
display portions of several files in his display  
areas (maintaining separate view control  
parameters for each display area) and may freely  
edit across the display area boundaries. The user  
may also have a list of frozen statements (from  
any currently open file) associated with each  
display area.

6clb2a

Initially, a user with a typewriter-type terminal  
will continue to have only one file and one set of  
viewspecs.

6clb2b

New String Processing Routines

6clb3

A new set of string manipulation routines was  
added, as well as new string constructs in the  
L-10 language which allow the use of string  
mechanism from a higher level.

6clb3a

Input Specification Constructs in L-10

6clb4

Constructs are being added to L-10 which make it  
easy for a user to specify personal commands.

6clb4a

The same constructs will facilitate the  
description and implementation of the NLS  
command language.

6clb4al

Context Group

6clb5

The user will be able to limit the sequence  
generator to a particular group within the file.  
This mechanism allows the user to restrict his  
activities to a portion of a file.

6clb5a

Modified or Deleted Capabilities:

6clc

Structure Manipulation	6clc1
These routines were modified to allow for cross-file editing.	6clc1a
Statement destruction	6clc2
The routines that remove statements were modified to combine free space in the statement data blocks and so allow better use of this free space by the statement construction routines.	6clc2a
Statement Construction	6clc3
Statement construction routines were modified to make better use of the free space in a statement data block, to make use of the capability of L-10 to manipulate parts of a word (called fields), and to allow for string construction in string buffers as well as statements.	6clc3a
Text Editing	6clc4
These routines were modified to allow for editing of strings as well as statements.	6clc4a
Literal Feedback	6clc5
The literal feedback mechanism was completely rewritten to allow for multiple display areas.	6clc5a
Input Feedback Support	6clc6
The input feedback support routines were modified to make use of fields in L-10, and to make the routines more consistent.	6clc6a
NLS Input Routines	6clc7
Character input routines were reorganized with the more basic routines modified to account for the TENEX system.	6clc7a
Markers	6clc8
Markers were called pointers on the 940 system. A marker is a symbolic name which the user may attach to a particular character in a file. Use of markers was restricted to the file being	

displayed in the 940 system, but we modified the lookup routines to allow reference to all of the files which are currently open (this modification may not be used initially). 6clc8a

Calculator System 6clc9

The calculator system was modified to make use of the double-word arithmetic instructions of the PDP-10. 6clc9a

Substitute 6clc10

For the user, the editing command "substitute" operates as it always has, but internally it was completely rewritten and reorganized. 6clc10a

Output Processor 6clc11

The output processor on the PDP-10 will be similar to the output processor (PASS4, which prepares files for printing and other graphic reproduction) now available on the 940 with the addition of new directives and a TREE-META-generated directive recognizer. 6clc11a

Insert Sequential 6clc12

The insert sequential facility was expanded to incorporate the insert QED function of the 940 system. The change decreases command execution time considerably. 6clc12a

Content Analyzer-Analyzer Compiler 6clc13

The analyzer compiler is replaced by the L-10 compiler, which now includes the capabilities of the FFL analyzer compiler. The content analyzer also will make use of the L-10 compiler. 6clc13a

File Compactor 6clc14

Used in the process of outputting a file, this facility was completely rewritten to make use of the multiple file capabilities of NLS/TNLS. 6clc14a

File Input/Output 6clc15

The Load File, Output File, Load (more recent,

older) Checkpoint, Output Checkpoint commands are either new or completely rewritten to account for the new file system. Automatic checkpointing has been added. 6c1c16a

Initialization 6c1c16

Parameter specification 6c1c16a

These routines were almost completely rewritten to take advantage of the added capabilities of L-10. 6c1c16a1

Sequence Generator 6c1c16b

The sequence generator was partially rewritten to make possible desirable changes in the sequence generator work area and to allow for the 'SEND' feature by making it a co-routine.

6c1c16b1

Frozen Statements 6c1c16c

Frozen statements are handled as they were on the 940, except that frozen statements may be associated with each display area and that the frozen statement lists may contain statements from any file currently open. 6c1c16c1

Verify (cleanup) 6c1c16d

A command to verify a file replaces a command to clean up a file. Verification is a fast read-only inspection of a user's file. 6c1c16d1

Bug Selection 6c1c16e

The routines which use the position of the cursor to determine a location within a file being displayed (the bug selection routines) were modified to be compatible with multiple display areas. 6c1c16e1

Display Generation 6c1c16f

The display image generator was entirely rewritten and recognized to allow for 1) control of the display by the TENEX monitor, an IMLAC display-processor, or a host computer via the ARPA Network, 2) multiple display areas,

and 3) eventual replacement by the portrayal generator. 6clc16f1

It now creates a universal display image. Device dependent secondary processors convert the universal display image to something compatible with the user's device. 6clc16f1a

Initialization routines were almost entirely rewritten to be compatible with the TENEX system. 6clc17

Message Display 6clc18

Modified to allow for addition of messages to extant messages on the screen and for multiple display areas. 6clc18a

String Routines 6clc19

Extant string manipulation routines were rewritten to make use of the PDP-10's byte manipulation. 6clc19a

Text Pointers 6clc20

The use and implementation of the text pointers were changed to allow pointers to point to the gap between characters (interstitial) rather than to one of the characters. This greatly simplifies their use. 6clc20a

Text Editing 6clc21

The basic text editing routines were rewritten to implement interstitial text pointers and be compatible with the L-10 language. 6clc21a

TNLS Input 6clc22

The most basic routines were rewritten to be compatible with the TENEX system. 6clc22a

TNLS Command Specification 6clc23

The TNLS command specification was partially rewritten and reorganized to allow for changes and reorganization of the support routines and to be more (structurally) similar to the NLS command specification. 6clc23a

File Manipulation	6clc24
The ring and data block manipulation, core page status table routines, and so forth, were extensively rewritten to take advantage of a more powerful file system.	6clc24a
Character Readout	6clc25
The routines that read characters from strings were modified to use the capabilities of the L-10 language, the PDP-10's byte manipulation instructions, and to read characters from strings as well as statements.	6clc25a
NLS Command Specification Routines	6clc26
The main NLS control routines--command language parser--were rewritten to conform to the replacement of the SPL language by L-10 and were reorganized to allow the user access to the parameter specification segment and the command executor segment of each command.	6clc26a
Data	6clc27
The writeable data declarations are almost completely new. we now use local variables when appropriate and the renaming of unclear global variables.	6clc27a
Keyword System	6clc28
The keyword system will be replaced later by a more powerful associative searching tool.	6clc28a
Trails System	6clc29
The trails system will be replaced later by a more powerful associative searching tool.	6clc29a
Tree Display	6clc30
The principle of bootstrapping forced us to delete tree display from the system because it was little used.	6clc30a
Merge file (filtered copy)	6clc31

SRI-ARC 1 JULY 71 0277  
Changing from an XDS-940 to a PDP-10

A command similar to that available now on the 940  
is provided in a cleaner and safer manner. 6clc31a

Don't Modify Working-Copy 6clc32

A capability similar to that available now on the  
940 is being provided in a cleaner and safer  
manner. 6clc32a

Collector-Sorter 6clc33

At the end of 1970 we had not as yet determined  
whether this feature will be provided as now  
available on the 940 or incorporated into NLS  
itself as a set of new commands. 6clc33a

Graphics Package 6clc34

A new graphics system (also available to the  
calculator compiler) includes a new data  
structure, "boxes", "areas", and normal editing of  
labels. 6clc34a

Execute Text for Display Oriented NLS 6clc35

An execute text command will be provided for NLS,  
if the programming and decreased efficiency is not  
too expensive. 6clc35a

VII. NEW FEATURES IN 1970

7

A. NEW TOOLS FOR USERS.

7a

1. During 1970 we developed the following substantial new  
features for users:

7a1

Collector-Sorter

7a1a

The Collector-Sorter is an NLS/TNLS subsystem which  
operates on a list of NLS files supplied by the user  
to extract statements that pass some user-specified  
content analysis program. The program may reformat  
the statements, and the Collector-Sorter may sort the  
collected statements with respect to specified  
"keys", which are appended to the statement by the  
content analysis program. It places the statements  
on the first level in a series of NLS files named \*1,  
\*2, ..., where \* denotes a name given by the user. 7a1a

Mail system

7a1b

The Mail subsystem allows one to send messages to  
other users and simultaneously submit the messages to  
the Journal. The Mail is available as a normal  
subsystem, and also is automatically queried when a  
user enters NLS/TNLS. If the user has no messages  
pending, he goes directly into NLS/TNLS. Otherwise,  
he is informed of the pending messages and is left in  
the Mail subsystem, with termination taking him into  
NLS/TNLS. While in the Mail subsystem, the user may 7a1b1

query the number of messages,

7a1b1a

query who sent the messages, when, and what the  
message journal numbers are,

7a1b1b

have the messages typed at his terminal or put  
into a file,

7a1b1c

have them simultaneously typed and deleted,

7a1b1d

delete any or all messages,

7a1b1e

and send messages to other users

7a1b1f

by either typing them at the time of sending or  
by naming a file from which the message(s) are  
retrieved.

7a1b1f1

Analyzer Compiler 7alc

The language which was developed for use in the specification of text entities and text editing algorithms was made available to the users. This language allows any user to develop very complicated personalized text editing. The Analyzer Compiler has been extensively used for the Network Information Center catalog management.

7alc1

Executable text in TNLS 7ald

The Execute Text command interprets an NLS statement as a string of input characters, just as though the user had typed them as command specification. A comment mode and a switch character, to switch from normal keyboard input to executable text input, are provided. This feature provided the first stage in the development of higher level capabilities in NLS/TNLS.

7ald1

Calculator and Calculator Compiler 7ale

The new calculator and calculator compiler replaced and expanded the earlier calculator. This new NLS subsystem allows users to do simple arithmetic operations on numbers in NLS files as well as to write programs to do more complicated analysis. The algebraic (Tree Meta produced) language provides constructs which elicit user responses, such as selection of a number in the file or the name of a procedure, variable, or calculator accumulator.

7ale1

Cross reference facility 7alf

The cross-reference facility allows the system programmers to produce cross-reference listings for their NLS source files.

7alf1

Execute Merge 7alg

The Execute Merge command allows the user to transfer all or part of one NLS file to another while retaining its hierarchic structure (when possible) and invoking various statement selection mechanisms such as level clipping or content analysis, if desired.

7alg1

Substitute 7alh

Network Information Center  
and Computer Augmented Team Interaction

The substitute command allows one to replace one set of text strings by another throughout a structural entity, invoking statement selection mechanisms if desired.

7alhl

Transpose command

7ali

The transpose command allows one to interchange two entities (strings of characters, statements, or groups of statements, in an NLS file.

7alil

Bug selections in replace command

7alid

The replace command in NLS was expanded to allow optional selection of the replacement entity by means of the cursor.

7alj1

Output processor directives

7alk

The Output Processor is an NLS file formatter, driven by embedded directives, for various output media, such as printer and microfilm. This NLS subsystem was expanded to incorporate several new directives (to simplify report production) and to initialize several directivees from the setting of the viewspecs at the time the output request was made. This report was produced using these new directives and the output processor.

7alkl

Quickprint

7all

Quickprint gives the user a very quick print out of all or part of an NLS file. Unlike the output processor, quickprint ignores embedded directives and formats strictly according to the viewspecs at the beginning of the quickprint. Statement selection mechanisms such as content analysis can also be used.

7alll

Character translation in TODAS

7alm

An expanded set of viewchange commands implemented user control of character set translation as described above. In addition, it allows the user to define various shift characters, set the number of rows and columns to print on a page, set the page size, set tab stops, and save his definition in a file.

7alml

Jump to Content and Jump to Name

7aln

The Jump to Content command scans statements for the string which was entered or selected by the user. If found, this statement becomes the new display-start statement, that is, the statement to which the Current Statement Pointer (CSP) points (note that the content analyzer may remove this statement from the display image). The qualifiers 'First' and 'Next' specify that the scan should begin at the origin or at the statement following the current display-start statement, respectively. These qualifiers also may be used with the Jump to Name command. 7alnl

Insert/Output Sequential 7alo

These commands convert NLS (random) files to sequential files and vice versa. 7alol

Execute TNLS/NLS 7alp

Allows the user to freely move from NLS to TNLS if he is at a display terminal. 7alpl

New Viewspecs 7alq

Two new statement selection viewspecs were added: 7alql

1) Plex only: restricts the sequence generator to the plex of the source of the display-start statement 7alqla

2) Content Analysis Fail: allows the sequence generator to select only those statements which fail to pass the current content pattern. 7alqlb

Reset File 7alr

Allows the user to discard his current file and revert to a null file. 7alrl

In addition to the above, we wrote new user's guides for NLS/TNLS, the output processor, and the calculator. 7als

B. CORE NLS 7b

1. As NLS has evolved, it has become apparent that a rational approach is needed to formulate it so as to be usable from a large diversity of terminals. It further became apparent that it would be desirable for a large number of diverse processes to have access to the NLS file

and text manipulation machinery. We have developed a new concept of the NLS program structure to provide these capabilities.

7bl1

In this concept, a central collection of NLS routines serves as a library for all of the basic functions of NLS. Included among these basic functions are File Handling, Structure Manipulation, Text Editing, and other functions which are useful for NLS programs. There is then a collection of processors or front ends, which are free to call on any of the routines in the Core NLS library. We call this library "Core NLS". As this model is evolved, the processors which call directly on the Core NLS routines become in fact trees of processors, with the following conventions:

7bla

The lowest node in the tree is that node which calls only on Core NLS routines. Any higher node may invoke any of the Core NLS functions, in addition to any higher level functions that are provided by nodes lower than itself, and in the same lineage. All terminal nodes on a tree are, in the terminology used above, processors for the NLS system.

7bla1

These processors may now share common libraries, which are represented by lower nodes on the tree. E.g., all processors which deal with a certain type of display could share the library necessary for driving that display. Transportation between terminal nodes on the tree allows a processor at one terminal node to pass control to a processor at another node (e.g., as TNLS may be called from NLS).

7bla2

There are two forms of calls: one is actually a branch, or a non-returning call, and the other corresponds to a procedure call in ALGOL. In this second case, parameters may be passed from the first processor to the one being called, and a processor may return a value. A stack is used to keep track of the return information and parameters. The stack allows recursion in the calls.

7bla2a

NLS (as a user system), TNLS, the Calculator, and the Collector/Sorter are examples of processors using Core NLS.

7bla3

Further development of the model will turn the tree into a network of nodes where each node may serve a

Network Information Center  
and Computer Augmented Team Interaction

processor function and a library function. As a processor, each node may perform a specific (set) of tasks which may or may not interact with a user. As a library, any node may be invoked by any other node, and then perform either a specially defined library function, or the function it would normally perform as processor.

7b1b

We are now making the necessary changes in the NLS System; the final reorganization in net form should be complete in June 1971.

7b1c

#### C. NEW HARDWARE TOOLS

7c

1. Three significant hardware changes in addition to the new computer during the past year were (1) the addition of UNIVAC drums for a swapping medium, (2) the addition of several new types of typewriter terminals and (3) the addition of an IMLAC Display terminal.

7c1

##### 2. UNIVAC Drums

7c2

In late 1969 we made a fairly extensive study of factors affecting response time in the 940 system. Based on this study the decision was made to replace the drums in use on the 940 with higher speed drums in the hopes of significantly improving response.

7c2a

The drums were connected to the 940 through a second memory interface connection and an interface designed and built to ARC specifications.

7c2b

The UNIVAC drums operated through a UNIVAC controller designed to operate with an 1108 system. The interface was therefore required to make the 940 look like an 1108 to the drum system.

7c2b1

In a manner similar to that used in many other 940 peripherals, a command table is stored in 940 core, giving all information relative to the transfer, including drum address, core address, word count, direction, and type of transfer required. The interface reads this command table and stored word count and core address in its own registers. The drum address and type of transfer requested are used to make up a 36-bit function word which is transmitted to the UNIVAC controller.

7c2b1a

The interface also converts 940 positive logic to the negative logic of the UNIVAC system and performs 24-to-36 bit conversion by packing one and a half 940 words to each UNIVAC word. 7c2t1b

Switch over to the UNIVAC drums led to a significant quickening of response. Although no actual measurements were made, our general feeling is that the predictions based on response studies were fairly accurate and that we got the improvement we expected. 7c2c

Our experience with the UNIVAC drums' reliability has been very good, and UNIVAC maintenance and field service are excellent. 7c2d

### 3. New Terminals 7c3

In the past year many new typewriter terminals for remote computer access have come on the market. These have been designed for many applications and use with many different systems, but very few met our requirements: 7c3a

Upper and lower case alphabet with a full complement of ASCII control codes; 7c3a1

Full duplex operation; 7c3a2

Character rate of at least 15 and preferably 30 characters per second; 7c3a3

In addition to these specific features, we look for quiet, reliable, small, light terminals with reasonably good print quality and generally desirable appearance. 7c3a4

These features, particularly upper and lower case alphabet, eliminate most of the available terminals. 7c3b

The terminals in use at ARC by the end of 1969 included Model 33 teletypes, Model 37 teletypes, G-E Termi-Net 300's, and Execuports. (ref.3) 7c3c

Of these terminals, all are still in use with the exception of the G-E Termi-Nets. Maintenance problems and the generally low reliability of these terminals forced us to cancel our lease. 7c3c1

Of the others, the Model 33's are generally the

stand-by for system use, monitoring teletypes, etc.  
because of their low cost and familiarity. 7c3c2

At the end of 1970 we were still using Model 37  
teletypes, but did not consider them desirable  
because they are large and noisy. 7c3c3

The Execuports are still highly satisfactory as  
portable terminals and have needed no maintenance  
whatsoever. 7c3c4

The only new terminal put into service in the last year  
is the Texas Instruments Model 720. Five of these had  
been in service for approximately one month at the end  
of 1970, and so far our experience had been very good. 7c3d

#### 4. IMLAC Display System 7c4

For some time we have hoped to incorporate a  
medium-speed remote display terminal as part of the  
facility and to experiment with using this terminal both  
as a high speed typewriter and as a modified display NLS  
terminal. 7c4a

Early last year the IMLAC display system was introduced.  
It is attractive in price and seemed to have many of the  
features we were looking for in an experimental  
terminal. 7c4b

The IMLAC is a small 16-bit machine with an  
arithmetic processor and a display processor  
operating from the same memory. The display  
processor drives a 9- by 11-inch display tube mounted  
in a separate unit. Input in the standard unit is  
from a keyboard that is read by the arithmetic  
processor and communication is through full duplex  
EIA interface. 7c4b1

For the IMLAC to operate as a remote NLS terminal it  
was necessary to add a mouse for display selection  
and keyset such as that used in the local display  
terminals. ARC personnel added them in a  
straight-forward manner. 7c4b2

Mouse coordinates (8-bit) for X and Y directions  
are ready by an I/O instruction into a single  
16-bit IMLAC word. The second I/O instruction  
reads the state of the five keyset switches and  
the three mouse switches. Software in the IMLAC

tracks the mouse position from the screen,  
interprets the mouse switches, and provides an  
algorithm for interpreting the five-finger keyset  
output as characters. 7c4b2a

The IMLAC is currently operating at 2000 baud over a  
Bell System 201A data set at a remote location. The  
data set connects at the ARC end to a data set  
controller operating from the 24-bit external core  
system (see Figure 1). 7c4c

D. HIGHER LEVEL PROCESSES 7d

1. During the past year we have expended considerable  
resources in the development of tools for extending our  
higher-level process capabilities. 7d1

By "higher-level processes" we mean processes in which  
the basic user-features of our online systems  
(particularly NLS) are used as "building-blocks" in the  
construction of programs for carrying out specific,  
perhaps rather complicated tasks. 7d1a

HLPs are in general used to automate text processing  
operations which, by virtue of frequent use, are too  
repetitive and time-consuming to do by hand. 7d1b

One of the major users of these higher-level process  
(HLP) tools has been the Network Information Center,  
which has utilized many HLPs in managing, searching, and  
print-formatting the NIC collection catalog as well as  
in other task areas. 7d1c

Four principal HLP tools are described below. 7d1d

2. Content Analyzer 7a2

Introduction 7d2a

The Content Analyzer (CA) feature of NLS permits the  
user to write, as part of any file statement, a  
string of text which specifies in a special language  
some pattern or content. 7a2a1

After the pattern has been compiled, whenever the  
content analyzer is turned on (through the use of  
a VIEWSPEC parameter) only statements that satisfy  
the content specification will be displayed,

printed, output, or affected by "Substitute" commands.

7d2a1a

If the user chooses (through use of a different VIEWSPEC parameter), only statements not satisfying the content criteria will be passed. 7d2a1b

The pattern specified may be simple -- e.g., a string of characters that may appear anywhere in a statement -- or complex -- e.g., a string, followed within a given number of words by another specified string, in statements created after a certain date by a certain author, and not containing some third specified string.

7d2a2

The language for specifying content patterns is simple and easy to use for simple cases, but powerful enough to be useful in more complex cases as well.

7d2a2a

#### The Process of Searching a Statement

7d2b

When the Content Analyzer is turned on, each statement in the file is searched, character by character, for the content specified in the pattern. Normally, the search begins with the first character, but it is possible to cause the search to proceed backwards from the end of the statement.

7d2b1

The CA uses a pointer to keep track of the search. The pointer always indicates which character is to be examined next, unless something in the pattern causes the pointer to be moved first.

7d2b2

At any given moment in the search process, the analyzer is searching for one of four types of content entity:

7d2b3

A literal string of characters, such as "abcd" or "13-x" or "ed Mat" or "memory."

7d2b3a

A string of "character-class variables" specifying, for example, "three digits, one after another," or "two letters, followed by any number of spaces, followed by three to five letters or digits."

7d2b3b

The date associated with the statement. (This is not normally printed or displayed as part of the

statement text, but every statement bears user-accessible data specifying the date on which it was created or most recently modified.) '7d2b3c

The initials associated with the statement. (As with the date, patterns may test the initials of the user by whom any statement was created or most recently modified.) '7d2b3d

All of the more complex analysis is achieved by moving the pointer according to the logic of the pattern specification. '7d2b4

For example, if the analyzer is to start at a given point and find either String A or String B, it first looks for String A; if String A is not found, the pointer is returned to the starting point, and a search is made for String B. '7d2b4a

### 3. Analyzer-Formatter

'7a3

The Content Analyzer is an old HLR, having been an integral part of NLS for several years. During the past year an expanded version of the CA, called the Analyzer Formatter (AF), has been incorporated into NLS. The AF permits the use of more complicated filtering patterns and also provides capabilities for reformatting or "programmed editing" of text statements. '7d3a

The Analyzer-Formatter is used in much the same way as the Content Analyzer, the major difference being that the AF has far more flexibility and power than the CA, and consequently, requires that a user master a more complicated language for specifying patterns. '7a3a1

Whereas CA patterns are restricted to being short strings of text, AF patterns are specified in an algorithmic language that permits powerful tools such as conditional statements and subroutine calls to be used in describing how a statement is to be searched and altered by the Analyzer-Formatter. '7d3a1a

In spite of this power, however, the AF is easy enough to use that sophisticated users frequently write AF programs for one-time use in editing specific NLS files. '7a3a1b

The AF has been heavily used in the conversion of

SRI-MRC 1 JULY 71 0277  
New Features in 1970

catalog files from old formats into a single new format  
and in processing the internal text codes into more  
readable forms for human consumption.

7d3b

The statements below are, respectively, the text for  
a single catalog entry as it appears in a master  
catalog file and the text produced by reformatting  
selected parts of this entry for inclusion in a  
"shelf-list" for online viewing and hardcopy  
printing:

7d3b1

<version 1>

7d3b2

(A5474) \*al Richard S. Marcus \*a2 Alan R. Benefeld \*a3  
Peter Kugel #2 Massachusetts Institute of Technology #3  
Electronic Systems Laboratory #5 Cambridge,  
Massachusetts \*cl The User Interface for the Intrex  
Retrieval System #6 42p. \*d1 (January 1971) \*d4 14-15  
January 1971 \*f1 d p \*f2 c \*ml AFIPS Information Systems  
Committee #1 The User Interface for Interactive Search  
of Bibliographic Data Bases, workshop #5 Palo Alto,  
California \*n1 5468 \*n6 5469 5470 5472 5473 5475 5476  
5478 5479 \*s1 National Science Foundation \*s2 Council  
for Library Resources \*s3 Carnegie Foundation \*w1 1-5-71  
\*w2 1-13-71 \*w3 dce \*w4 John L. Bennett #2 IBM Research  
Laboratory #3 Information Sciences Department #4  
Monterey & Cottle Roads #5 San Jose, California 95114  
\*yl Describes decisions made in design of system/user  
interface for Intrex, grounds for decisions, and results  
obtained by experiments with users. Finds high degree of  
user acceptance as implemented. Indicates desirable  
improvements. \*z2 AtI \*z3 new \*

7d3c

<Version 2>

7d3d

The User Interface for the Intrex Retrieval System (Draft) 5474

Richard S. Marcus, Alan R. Benefeld, and Peter Kugel  
(Massachusetts Institute of Technology, Electronic  
Systems Laboratory, Cambridge, Massachusetts).

(January 1971).

Describes decisions made in design of system/user  
interface for Intrex, grounds for decisions, and  
results obtained by experiments with users. Finds  
high degree of user acceptance as implemented.  
Indicates desirable improvements.

7d3e

Network Information Center  
and Computer Augmented Team Interaction

4. Collector-Sorter

7d4

The Collector-Sorter (CS) is a subsystem called from NLS that automates the process of collecting statements from one or more NLS files and sorting them into one or more new files.

7d4a

The Collector-Sorter is usually used in conjunction with an Analyser-Formatter program, so that in the collection process statements may be arbitrarily reformatted by the AF program. The AF program can also be used to select from the text of each statement strings to be used as sort keys for that statement.

7d4al

The Network Information Center has made heavy use of the CS in preparing hard-copy catalogs and shelf lists from the machine-readable master NIC catalog.

7d4b

5. Executable Text

7d5

The Executable Text (ET) feature of TNLS is an early attempt to provide users with an easy-to-use procedural language for manipulating information contained in NLS files.

7d5a

This feature permits users to request that some body of text within a file be interpreted as if it were the user's own keyboard input stream.

7d5al

ET commands may be used to perform any NLS editing operations, including changing the ET "program" itself. They may also be used to perform file-manipulating operations, such as loading, updating, and printing, and it is possible for an ET program to link to another ET program in a different file.

7d5a2

Executable Text alone can be used to automate simple file editing operations, and in conjunction with the AF and CS it provides users with a powerful mechanism for writing programs to perform complex editing tasks as well as some forms of user-interaction.

7d5b

E. DESIGN TEAM AUGMENTATION

7e

1. The Need

7e1

ARC has become more and more involved in augmentation of

Network Information Center  
and Computer Augmented Team Interaction

teams, and we are giving serious consideration to improving intrateam communication with whatever mixture of tools, conventions, and procedures will help. 7ela

If a team is solving a problem that extends over a considerable time, the members will begin to need help remembering some of the important communications -- i.e., some recording and recalling processes must be invoked, and these processes become candidates for augmentation. To consider some of the different conditions where such storage and recall may be useful, suppose Person A communicates with Person B about Item N at Time T. 7elb

They may well be counted on to remember their exchange during the problem-solving period. But consider the case of Person C who, it will turn out, is going to need to know about this communication at Time TT: 7elbl

Perhaps he was there at Time T but, 7elbla

he was too heavily involved even to notice the communication, and/or Item N wasn't relevant to his work at that moment and so wasn't implanted for ready recall. 7elblal

Perhaps A and B didn't anticipate his later need and thus failed to invite him into their interchange or inform him of its conclusion. 7elblb

Perhaps, although Persons A and B knew he would later need the information, they didn't want to interrupt their own working sequence with the procedure of interrupting Person C and getting him involved. 7elblc

Or, if the consequences of the interchange carry over into a long-lasting series of other decisions, one or both parties may fail to remember accurately, or may remember differently because of different viewpoints, and troublesome conflicts and waste of effort may result. A single person will make a list of things to do on a shopping trip because he's learned that the confusion and pressure may make him forget something important. It's obvious that to be procurer for one of a mutually developed, interdependent pair of lists would make it even more important to use a record. 7el02

Further consider the effect if the complexity of the team's problem relative to human working capacity requires its partitioning into many parts where each part is independently attacked, but where among the parts there is considerable interdependence through interactions on mutual factors such as total resource, timing, weight, physical space, functional meshing. 7elc

Here, the communication between Persons A and B may well be too complex for their own accurate recall. For example their communication period resulted in scratch paper or a chalkboard covered with possibilities and the essence of the agreed-upon solution which has since disappeared. 7elcl

We envision effectively augmenting our collaborative team by having an "intragroup documentation system", containing current and thoroughly used working records of the group's plans, designs, notes, etc. Therefore, we have begun to develop a system for entering and managing those records. The ARC Journal is this intragroup documentation system. 7eld

## 2. The ARC Journal 7e2

Our Journal is an open-ended information storage and retrieval system. It accommodates and retrieves whatever thoughts any member of the group feels worth keeping. All entries in our internal "mail" system automatically become part of the Journal. In addition, any online user may flag any file for transcription into the Journal within a day. In addition to NLS files, other hard copy including photographs, line drawings, and scratch notes can be logged into the Journal. In handling extra-computer copy the Journal draws on the techniques we are developing for NIC and KINS. In this section of this report, we concentrated on the Journal as recipient of NLS files. 7e2a

We believe the Journal is the key to the development of our Dialogue Support System. We are encouraging members of the group to enter items freely, to err on the side of loquaciousness, even to enter information that will become useless. We hope to learn from such a flow how to winnow worthwhile information, to refine the techniques of query, analysis, and access that are necessary to proliferate all our augmentation research. 7e2b

As each item (in this case, every NLS file) enters into

the Journal it receives master Catalog Number (CNUM) and  
is catalogued. 7e2c

The CNUM is generated from the one master-collection  
sequence that ARC uses for all of its frozen-item  
storage: XDOC, NIC, Journal, RINS, and, we assume,  
an increasing number of other special collections.  
The CNUM becomes the master identifier of the NLS  
file: it is printed in the upper right corner of each  
page of a printout of that file; it is the standard  
reference name to use in an NLS link; and it becomes  
the "file name" of that file within the storage and  
retrieval system of the Journal. 7e2cl

When the Journal System takes a file into custody, it  
guarantees retrieval of that file (by its CNUM) at any  
later time. 7e2d

A Master Catalog holds descriptions of each item that is  
stored in ARC's Master Collection. The Master Catalog  
is composed of a set of NLS files in which each entry  
(describing one collection item) occupies one statement  
whose NLS name is 'M+CNUM -- e.g., (M5237) 7e2e

The catalog entries are formatted in a special way to  
delimit the different data elements. For instance,  
for most items there is a "\*al" preceding the  
first-author's name, and within this type of main  
field there often are flags such as "#2" or "#3" to  
delimit a particular subfield. The initials of the  
ARC author are stored after the data element code  
"\*a6". 7e2el

We don't really expect to use this format  
permanently for storing our catalog data. Within  
a year the size of the collection will make query  
and file management operations too inefficient and  
we will change it. A collector sorter and special  
reformatting programs will reduce the work of  
designing and changing the new format to several  
hours at the console. 7e2el1

The organization and formatting of the catalog  
files will evolve during the next year, but the  
user's concept of this function probably won't be  
affected. 7e2elb

Special data elements are under consideration for  
processing our NLS files into the journal. For

instance, it is likely that the catalog entry will invoke a record of the whereabouts and the reference target of every cross-file link with the file. Such a notation would be an important aid in querying and is also the base for the "back-linking" we have been considering for so long. 7e2e2

Journal entries now also exist as a shelf of hard copies. For the shelf-stored copies we now have what we call "catalog-management processes", (EXecutable Text) Programs to help manage and retrieve the information. 7e2f

The catalog-management techniques that we have used were designed expressly to accommodate special collections. For example, a working subset of the Master Catalog holds the Catalog entries for the items that have been entered in the Journal. This subset is called the "Journal Catalog", and can be extracted automatically from the Master Catalog. Our initial shelving is by Catalog number, so the shelf list is by CNUM. 7e2f1

Initial Journal catalog format: 7e2f2

(M4898) \*a6 DCE \*c1 Comments on WSD 4897, Catalog Query System \*a6 10/22/70 \*a7 0955:25 \*f3 :JRNLA \*z2 JOU \*z3 new \* 7e2f2a

(M4899) \*a6 WKE \*c1 10ACQ \*a6 10/22/70 \*a7 1027:25 \*f2 :10ACQ \*z2 JOU \*z3 new \* 7e2f2b

(M5200) \*a6 VDB \*c1 New NLS Calculator \*d6 10/30/70 \*d7 1140:45 \*f2 \*CALDOC \*z2 JOU \*z3 new \* 7e2f2c

(M5201) \*a6 MAIL \*c1 MAIL FILE \*d6 11/04/70 \*d7 1015:52 \*f2 :MAIL \*z2 JOU \*z3 new \* 7e2f2c

(M5202) \*a6 DCE \*c1 Old but Relevant NIC Notes from Aug 70 \*a6 10/29/70 \*d7 0911:26 \*f3 :JRNLA \*z2 JOU \*z3 new \* 7e2f2e

(M5203) \*a6 WLB \*c1 ENTRY TO NIC LIAISON LOG - WLB-UCSB \*a6 10/29/70 \*a7 1111:11 \*f3 :LIAISON LOG \*z2 JOU \*z3 new \* 7e2f2f

(M5204) \*a6 WLB \*c1 ENTRY TO NIC LIAISON LOG - WLB-KAND \*d6 10/30/70 \*d7 1111:11 \*f3 :LIAISON LOG \*z2 JOU \*z3 new \* 7e2f2g

SKI-ARC 1 JULY 71 0277  
New Features in 1970

(M5216) \*a6 DVN \*c1 Meeting 11/2/70, DCE/DVN, JCN  
\*d6 10/06/70 \*d7 1541:56 \*f3 :DRAFT \*z2 JOU \*z3  
new \* 7e2f2h

(M5217) \*a6 WSD \*c1 Proposed New Features in  
Executable Text \*d6 11/05/70 \*d7 1131:24 \*f3  
:NEXTTEXT \*z2 JOU \*z3 new \* 7e2f2i

(M5218) \*a6 WSD \*c1 Proposed New Features in  
Executable Text, Revision 3 \*d6 11/06/70 \*a7  
1238:07 \*f3 :NEXTTEXT \*z2 JOU \*z3 New \* 7e2f2j

(M5219) \*a6 DCE \*c1 Requirements for higher-level  
interactive processes \*d6 11/06/70 \*d7 1630:00 \*f3  
:JRNLA \*z2 JOU \*z3 new \* 7e2f2k

We can automatically generate hard-copy citation  
lists in various layouts by means of a library of  
reformatting programs. The Collector-Sorter  
Processor is invoked in one set of executable text  
programs, to produce listings sorted on selected  
keys. 7e2f3

One such listing is the shelf list. A Shelf List for  
a given collection is a list of citations ordered in  
the way in which the collection items are physically  
"shelved" or otherwise stored. 7e2f4

Shelf list (by CNUM):

5208 DCE 11/04/70 Discussion Notes, DCE/JTM: Net access  
for NIC users  
Source: :JRNLA Time: 1303:33

5209 DCE 11/02/70 Some NP Notes on Analyzer Formatter  
and Executable Text  
Source: :ETAFL Time: 0918:42

5210 WLB 11/02/70 COMMENTS ON 5206 (PROPOSED EXECUTABLE  
TEXT FEATURES)  
Source: :MEMO Time: 0919:00

5211 MAIL 11/06/70 MAIL FILE  
Source: :MAIL Time: 1137:46

5212 WLB 11/03/70 ENTRY TO NIC LIAISON LOG - WLBRAND  
Source: :LIAISON LOG Time: 1108:07

Network Information Center  
and Computer Augmented Team Interaction  
51

5213 WLB 11/03/70 ENTRY TO NIC LIAISON LOG - WLB+UTAH  
Source: :LIAISON LOG Time: 1054:46

5214 DCE 11/05/70 Notes: DCE Talk with Rubin re. SRI  
Info-Sys Activity  
Source: :JMNLC Time: 0900:42

5215 MAIL 11/06/70 MAIL File  
Source: :MAIL Time: 1422:03

5216 DVN 11/06/70 Meeting 11/2/70, DCE/DVN, JCM  
Source: :DRAFT Time: 1541:56

5217 WSD 11/05/70 Proposed New Features in Executable  
Text  
Source: :NEXTTEXT Time: 1331:24

5218 WSD 11/06/70 Proposed New Features in Executable  
Text, Revision 3  
Source: :NEXTTEXT Time: 1238:07

If the items are standing on the shelf arranged by catalog number, you would probably find one easily without looking at the Shelf List. But, if the item is gone, the Shelf List can verify that it should be there.

The items might very well be shelved according to a subject outline -- e.g., a set of user-reference volumes whose sections would each be a separate journal entry. Here the various sections would be updated independently, and their catalog numbers would bear no relation to their ordering within the binders. The Shelf List here would look like a Table of Contents.

An "Index" contains one-line citations ordered alphabetically or numerically on one or more of the terms found in the catalog entries. We automatically produce indices ordered on: Catalog Numbers; Author; and Keywords from the title (having an entry for each non-trivial title word).

Author index (by initials):

5243 SLP 12/09/70 Partial Description of the Universal  
4860 CHI 09/11/70 New NLS features

Network Information Center  
and Computer Augmented Team Interaction  
52

5244 CHI 12/10/70 NOTES ON CHANGES TO THE NLS SYSTEM  
4803 DCE 06/03/70 Initial Journal System (Euitea version  
5219 DCE 11/06/70 Requirements for higher-level

Titleword index:

Word	CNUM	Auth	Date	Title (front only)
ACCESS	4832	WKE	07/10/70	NETWORK ACCESS TO SYSTEM
Access	4850	WKE	07/10/70	Network Access to System
access	5208	DCE	11/04/70	Discussion Notes, DCE/JIM: Net
ACCESSION	4869	WSD	10/06/70	PROGRAM FOR PRODUCING A TITLE
Activity	5214	DCE	11/07/70	Notes: DCE Talk with Rubin re.
Agency	4851	DCE	09/10/70	Setup of a National
AGENTS	5618	JBN	12/15/70	TRANSMITTAL TO NIC STATION
ANALYZER	5227	WLB	11/18/70	ANALYZER-FORMATTER PROGRAMS
Analyzer	5209	DCE	11/02/70	Some NP Notes on Analyzer
Answering	5228	JBN	11/20/70	Answering Service for the NIC
ANSWERING	5207	WLB	10/30/70	MEMO RE PALO ALTO ANSWERING SERVICE

7e2f'w  
We keep up-to-date copies of the Shelf List,  
Author Index, and Title-Word Index on the shelf  
beside the hard copies of the Journal. 7e2f'x

we will soon begin to divide the Journal into  
sub-collections, e.g.,: obsolete items; software  
documentation; Baseline Records; correspondence;  
etc. 7e2f'y

We plan to make journal material ever easier to read  
online. By next fall we hope that any NLS user studying

Network Information Center  
and Computer Augmented Team Interaction  
bj

a Journal item may jump from a link to any Journal item that has been referenced within the past few days with the speed of disc access, and with a "worst case" time of less than five minutes for a file not used recently. 7e2g

3. The Baseline Record:

7e3

The baseline Record is a special sub-collection of the Journal. It will consist of a series files specially formatted to contain task and resource allocation information, including files of plans, specifications, analyses, designs, etc. 7e3a

It will be composed of that portion of our current working records that represents our best definition of tasks we plan to perform in the future, how we are planning to do them, and what uses of resources (people, system service, materials) are expected. 7e3b

We will keep some or all of the Baseline Record within a specially organized subcollection of the Journal, shelved separately, and we will use as a "Shelf List" a topically organized Table of Contents. Sections of the Baseline Record that are superceded by new Journal entries will be retired to obsolete status. Changes will be approved and recorded as in configuration management of hardware designs. 7e3c

VIII. PLANS FOR 1971	8
A. NETWORK INFORMATION CENTER DEVELOPMENT AND OPERATION	8a
1. Computer and Network Use	8al
As necessary documentation becomes available, we will bring up the BEN Network Control Program (NCP) and BBN Telnet. We will then perform some testing before we provide network service.	8ala
Initially, our local connect capacity allows for 12 displays and 24 typewriter terminals. With about 10 displays and 6 typewriter terminals running NLS, response is satisfactory, but marginal for display users. The delivery in June of new Bryant drums and measuring and tuning the new system should increase capacity and response. How much improvement to expect is not known.	8alb
The system processing required to support a network user is heavier than required to support a local typewriter user. Therefore we are not sure how many network users we will be able to support without degrading response seriously or requiring that we limit local loading by administrative restrictions. Our initial hope is that we can handle 6 network users by mid-summer with an optimistic expectation that we might be able to handle closer to 12.	8ac
As there is only limited interactive experience over the network, we do not know what its response characteristics will be like. We may find that the delays caused by two timesharing systems and the network transmission may allow us to support the higher number of network users without adding serious incremental response delays. The loading caused by parallel processes controlling intersite file transfers is also an unknown factor at this point.	8ald
We plan to increase our reference and communication service capacity by providing deferred execution facilities which will allow NLS compatible file preparation and editing offline or in local hosts; files so created may then be entered into NLS for further manipulation.	8ale
To prevent file capacity from being inadequate when needed, we are studying ways of using tape or facilities	

such as those at UCSS to give us an integrated auxiliary facility. oalff

Our plans for providing online service to the network are briefly given below. oalg

Stage 0 (Mid-June): oalg1

Stage 0 is to provide experimental access to the NIC for RADC and a limited number of West Coast sites so that we can learn how to handle problems which may come up in actual network operation. These sites provide a variety of hosts and their location on the West Coast simplifies communication during this initial trial period. oalgla

Stage 0 will allow access to the TENEX Executive, TNLS, an initial Network Dialog Support System-DSS (which will allow online creation and submission of messages and documents, with harucopy mail delivery), and the first release of our TNLS users manual. oalg1b

Initially, we will allow a maximum of two network users on at once. oalg1c

There will be a two-day TNLS course at SRI in June for the initial sites. oalg1d

Stage 1 (Early August): oalg2

Stage 1 is to provide access to the NIC from any site in the network having the appropriate access software. oalg2a

Stage 1 will allow access to the DSS of Stage 0 with online access to documents and messages created online, online access to network related files such as the NIC Catalog, ARPA Network Resource Notebook, and other NIC documentation. oalg2b

We expect to provide training to sites desiring access. We will allow as many network users simultaneous access as we can, depending on initial success with system tuning. A reasonable guess is 4-8 users. oalg2c

Stage 2 (October): oalg3

Stage 2 will provide message delivery to files at remote sites (assuming protocols established by the Network Working Group have been implemented), an initial deferred execution mode allowing users to prepare files on their systems and then have them entered into TNLS for further work, and improved query facilities of network online files.

021g3a

We hope to have improved TENEX-NLS performance so as to allow more network users simultaneous access than allowed in Stage 1.

0a1g3b

## 2. Other Reference and Communication Activities

0a2

**Mailing:** We will continue to mail RFC's and other material going to liaison people as soon as we can get the material duplicated, which is usually within 24 to 48 hours after we receive it. We will mail material to station agents once each week, usually on Fridays. As online messages and documents are sent through the NICDSS, we will transmit copies to the addressees and to stations as appropriate.

0a2a

**Catalogs:** We will continue to produce NIC catalog listings and indices, using improved techniques for their formatting and printing. We will also develop more automatic procedures for handling the production of the catalog and maintenance of the master catalog citation data. Early design work and the production of the first catalogs have given us additional understanding of the problems involved and ideas for meeting these needs. We plan to produce catalogs on a monthly basis.

0a2b

## B. DIALOG SUPPORT SYSTEM DEVELOPMENT

0b

### 1. Automatic Journal Entry

0b1

After the transfer of NLS to the PDP-10, our Journal entry and cataloging procedures will be made more automatic, and brought under direct user control from NLS.

0b1a

Entry commands such as the following will be used:

0b1al

Execute Journal

0b1ala

Interrogate (optional interactive input  
request mode)

0b1alb

SRI-ARC i JULY 71 0277  
Plans for 1971

Author	(the user by default, others are entered)	0bblalc
Comments	(optional comments about the document)	0bblald
Distribution	(to ARC or non-ARC people by name)	0bblale
Subcollections	(NIC,AFIPS,NAS,etc.)	0bblalf
Keywords	(at user's discretion)	0bblalg
Expedite	(for 3-4 hour delivery to ARC addressees)	0bblah
GO	(to start file and catalog process)	0bblali

Catalog entry, hardcopy formatting, and secure online filing of the document are included in this process. 0bbla2

Hardcopy distribution will be used for all documents at first; optional online delivery to addressees of links (references) to the Journal document files will follow soon thereafter. 0bbla3

2. We plan to make Journal material ever easier to read online. By next Fall we hope that any NLS user studying a Journal item may jump from a link to any Journal item that has been referenced within the past few days with the speed of disc access, and with a "worst case" time of less than five minutes for a file not used recently. 0b2

3. Further development and detailed design of other needed DSS features including work on backlinking, set generation and manipulation, and comment handling will continue. 0b3

#### C. BASELINE MANAGEMENT SYSTEM DEVELOPMENT:

0c

1. The basic design and implementation of the ARC baseline management system will proceed with operational use of task planning procedures across various areas including development and operation in Service System, NIC, NLS, TENEX, Hardware, Dialog Support, File System, management System, and Documentation activities. 0cl

2. Task planning data collection will continue, with

Network Information Center  
and Computer Augmented Team Interaction

SRI-AKC 1 JULY 71 8277  
Plans for 1971

improvement to be made in methods of file updating by those responsible for task management.

8c2

key planning data elements include:	8c2a
requirements (what each task is supposed to produce)	8c2a1
buyer(s) (other task(s) sponsoring conduct of each task)	8c2a2
design details (or links to journal or other files)	8c2a3
milestone points (as appropriate)	8c2a4
estimated dates (start, completion, duration, milestones)	8c2a5
estimated resource use (people, system, other)	8c2a6
sub-tasks (as appropriate)	8c2a7
Dependencies on or by other tasks (by time or design)	8c2a8

D. TRANSFER OF NLS

8d

1. Transfer of existing NLS and TNLS features from the KDS 940 to the PDP-10 will be completed, with needed changes being made to those features where practical during the transfer process.

8a1

2. Key changes in TNLS will be made to give users more access to textual entities in viewing and editing operations. These will center about providing commands for specifying addresses more precisely and for movement of a control marker within a file to statements and within statements to character positions by character count, entity count, content, and other specifications.

8d2

3. TNLS changes will be made with the objective of giving network user access to NLS features and files in as useful a manner as possible, recognizing existing and future characteristics of the modes and terminals from which they will work.

8d3

E. NEW FEATURES IN 1971

8e

1. New NLS and Executive features planned next are those most directly supporting NIC development and operation tasks.

8el

2. Some Executive tasks are:

8e2

DRUM Diagnostics 8e2a  
Bryant-UNIVAC System 8e2b  
DRUM Comparisons 8e2c  
Disc Diagnostics 8e2d  
Disc Elevator Algorithm 8e2e  
NET Link and Advise Studies 8e2f  
Tertiary File Storage Study 8e2g  
Increase Open Files Capability 8e2h  
Network File Transfer Study 8e2i  
Performance Measurements 8e2j  
Study Capacity Increase Needs and Possibilities 8e2k  
Background Process Development 8e2l  
Reorganize NCORE 8e2m  
via Scheduling Design 8e2n

3. Some NLS tasks are:

8e3

Cross File Editing 8e3a  
Deferred Execution 8e3b  
Statement Address Options 8e3c  
Cross Reference 8e3d  
Statement Property Lists 8e3e  
One Command Background 8e3f  
Remote NLS Specifications 8e3g  
Command Backup 8e3h  
Collector Sorter Improvements 8e3i  
Fast Substitute 8e3j  
Portrayal Generator 8e3k  
Help Command 8e3l  
Novice Mode 8e3m

F. MODULAR PROGRAMMING

8f

1. A fully-developed augmentation system of a few years hence will have a very large repertoire of commands, representing a rich vocabulary for eliciting help from the computer system. To experiment meaningfully with any one subset of commands, designed to support a special kind of intellectual task, the evaluation must rightfully be done within a working environment in which the subjects are

doing all of their associated work in the way they would do it in the "complete workshop." of1

2. This means that to provide a progressive research environment in which rapid and significant evolution can take place, some sort of a "latest thing in complete workshops" must be maintained as a laboratory for each experimenter. To maintain this in separate installations is quite impractical. of2

3. The computer network offers an important hope here, in that it makes it possible for people at distributed locations to share a "latest thing in complete workshops" as an environment for their different, specific "tool-development experiments." of3

For several years ARC has been aiming toward an experimental future in which this was the way in which our work on augmentation systems would be done -- as part of a larger community in which many more people than we could marshal would be working on different fronts (and at different levels). of3a

For instance, much of our motivation toward the Dialog Support System has been to facilitate close collaboration between such distributed system-development participants. of3b

4. Besides being able to sustain collaborative dialog, the participants would be much helped if each could view a relatively stable system as the background in which he experimented with a new tool, and if he could very rapidly and independently create and modify new tool features. of4

5. We are launching development of a Modular Programming System explicitly to serve this end. Design and implementation of a preliminary system will occur during 1971 with further stages of development to follow. When NLS has been modularized, it will be possible for instance to permit a worker at Utah to be given "custodianship" of a private subset of modules pertaining to the manipulation of one kind of graphic-data packet in our file data nodes. of5

He would be given his private copies of the source code files for these modules, and could add and/or modify them at will. His modules could be independently compiled by him at any time; and when he wishes to experiment with the resulting "new tool," his compiled modules could be linked into the rest of the NLS

compiled-code module set at run time, perhaps in place  
of some modules that the standard version of NLS offers  
but that he is redoing.

8f5a

To experiment with his tool, he could use it in the  
midst of processes, methods and information that are  
part of a busy (and evolving) working life in the whole  
workshop.

8f5b

Each person could do his private development with  
minimal burden on the support system, and with maximum  
protection to the other workshop users.

8f5c

The standard-NLS Module Set would be controlled and  
updated by a central community process, steadily  
integrating the improvements of the trial tools as they  
become thoroughly checked out.

8f5d

IX. GLOSSARY	9
ARC -- Acronym for Augmentation Research Center.	9a
ARPA -- Acronym for the Advanced Research Projects Agency or the Office of the Secretary of Defense.	9b
Augmentation -- In this report, extension, improvement, or amplification of human intellectual and organizational capabilities by means of close interaction with computer aids and by use of special procedural and organization techniques designed to support and exploit this interaction.	9c
BB&N -- Bolt Beranek and Newman. A commercial research and development organization under contract to ARPA for services to the ARPA Network, and under other contracts that lead to frequent interaction with ARC.	9d
Bootstrapping -- A name for the research strategy of the ARC. By "bootstrapping" we mean taking advantage of the feedback in recursive development of systems. That is, we try to test ways of augmenting intelligence by their usefulness in developing new systems to augment intelligence.	9e
branch -- In the NLS hierarchy of statements, a statement and all substatements that depend on it.	9f
Center -- The same as ARC.	9g
Console -- As used here, specifically a user's control console for the ARC's Online System (NLS). The consoles presently in use consist of a display screen, a keyboard, a "mouse", and a "keyset".	9h
Current Statement -- In NLS, normally the last statement modified, executed, or reproduced by the user, and, hence the statement that starts the sequence of the sequence generator which generates the display image. Usually the statement at the top of the screen is the current statement, but content analysis or screen splitting may displace or obscure it.	9i
Current Statement pointer -- The internal symbol fixed on the current statement by NLS.	9j
Dialog Support System (DSS) -- The system of files, programs, and procedures at ARC for storing, sorting and recovering the interchange of thoughts, plans, memos, technical documents, etc. that accompany our system development.	9k

Display Start Statement -- The same as "current statement"	yl
Executable Text -- In NLS, a program or subroutine that is written in characters as all or part of a statement and that can be carried out by a simple command from the user.	ym
File -- In NLS, this refers to a unified collection of information held in computer storage for use with the online System. A file may contain text (English or program code), numerical information, graphics, or any combination of these. Conceptually, a file corresponds roughly to a hard-copy document.	yn
Field Operations -- In programming NLS, manipulations that involve the capacity of the PDP-10's software to handle parts of words.	yo
Frozen Statements -- In using NLS, statements held as is on the display while other parts of the file are composed or modified.	yp
Higher Level Processes -- (HLP) Processes in which the basic user features of our online systems (particularly NLS and TNLS) are used as building-blocks in the construction of programs for carrying out specific, perhaps rather complicated tasks.	qq
IMP -- Acronym for Interface Message Processors. Hardware devices that code and decode messages for transmission between the computers on the ARPA Network.	9r
Intellect -- The human competence to make, sort, exchange, and apply to decision making knowledge.	ys
Journal -- The open ended information storage and retrieval system that supports the Dialog Support System.	9t
Keyset -- A device like a stenographic machine consisting of five keys to be struck with the left hand in commanding the online system.	9u
List -- In the NLS hierarchy, the list of a given statement is the set of statements that are in the piec of the source of the given statement and are on the same level with it.	9v
Markers -- A marker is a symbolic name which the user may attach to a particular character in a file. It is invisible on the screen, but visible to routines that search for it.	9w

Mouse -- A device operated by the right hand in using the Online System. The mouse rolls freely on any flat surface, causing a cursor spot on the display screen to move correspondingly. yx

NIC -- Acronym for Network Information Center, ARC's key role in the ARPA Computer Network. The NIC is a computer-assisted reference and communication service for information pertaining to the network. yy

NLS -- Acronym for the ARC Online System. yz

Plex -- In the NLS hierarchy, the set of all statements that have a common source. ya\*

Online System -- This is ARC's principal and central development in the area of computer aids to the human intellect. As presently constituted, it is a time-shared multi-console system for the composition, study, and modification of files (see definition of "file"). Many details of the system are described in the body of this report. 9aa

Pointer -- An old name for marker. yab

RADC -- Acronym for Rome Air Development Center. yac

Sequence Generator -- A routine that, when given the number that identifies a statement internally (the STID), will search through the file and find all the subsequent statements that observe the current viewspecs. yad

SRI -- Acronym for Stanford Research Institute 9ae

STID -- Acronym for statement identifier. A number unique to each statement in a file and that remains with the data regardless of editing. yaf

Source -- In the NLS hierarchy, the first sublist of a statement is the set of statements immediately below it, the second sublist is all statements one level below them, and so the nth sublist of statement "s" is the set of statements that are in the first sublist of the statements in the (n-1)th sublist of "s". yag

Statement -- The basic structural unit of a file. A statement consists of an arbitrary string of text, plus graphic information. A file consists of a number of statements in arranged an explicit hierarchical structure. 9ah

Textpointer -- In NLS as used on the PDP-10, the fixation by NLS on a space between two characters which allows the users to be sure editing or execution of executable text will begin with the following character.

9ai

TNLS -- Acronym for Typewriter Online System. The system used in ARC from typewriter type terminals from early 1971 on. It differs from TODAS internally in using core NLS with adaptive routines that are called automatically when the user names his terminal in logging in, and externally in a number of additional powerful editing commands.

9aj

TODAS -- Acronym for Typewriter Oriented Documentation Aid System. The version of NLS used from typewriter like terminals prior to 1971.

9ak

Tree Meta -- The compiler-compiler system of ARC, used to compile all the languages at ARC.

9al

X REFERENCES AND PUBLICATIONS	10
REFERENCES	10a
1. D. C. Engelbart and Staff of Augmentation Research Center, "Computer-Augmented Management-System Research and Development of Augmentation Facility," RADC-TR-82, April 1970, Final Report of Contract F30602-68-C-0286, SRI Project 7101, Stanford Research Institute, Menlo Park, California.	10a1
2. D. C. Engelbart and Staff of Augmentation Research Center, "Advanced Intellect-Augmentation Techniques," Final Report NASA Contract NAS1-7897, July 1970, SRI Project 7079, Stanford Research Institute, Menlo Park, California.	10a2
ARC PUBLICATIONS	10b
1. D. C. Engelbart, "Special Considerations of the Individual As a User, Generator, and Retriever of Information," Paper presented at Annual Meeting of American Documentation Institute, Berkeley, California (23-27 October 1960).	10b1
2. D. C. Engelbart, "Augmenting Human Intellect: A Conceptual Framework," Summary Report, Contract AF 49(638)-1024, SRI Project 3578, Stanford Research Institute, Menlo Park, California (October 1962), AD 289 565.	10b2
3. D. C. Engelbart, "A Conceptual Framework for the Augmentation of Man's Intellect," in <i>Vistas in Information Handling</i> , Volume 1, D. W. Howerton and D. C. Weeks, eds., Spartan Books, Washington, D.C. (1963).	10b3
4. D. C. Engelbart, "Augmenting Human Intellect: Experiments, Concepts, and Possibilities," Summary Report, Contract AF 49(638)-1024, SRI Project 3578, Stanford Research Institute, Menlo Park, California (March 1965), AD 640 989.	10b4
5. D. C. Engelbart and B. Huddart, "Research on Computer-Augmented Information Management," Technical Report ESD-TDR-65-168, Contract AF 19(628)-4088, Stanford Research Institute, Menlo Park, California (March 1965), AD 622 520.	10b5
6. W. K. English, D. C. Engelbart, and B. Huddart, "Computer-Aided Display Control," Final Report, Contract	

Network Information Center  
and Computer Augmented Team Interaction  
67

NAS1-3988, SRI Project 5891, Stanford Research Institute, Menlo Park, California (July 1965), CRSTI Order No. N68-30204.

10b6

7. W. K. English, D. C. Engelbart, and M. L. Berman, "Display-Selection Techniques for Text Manipulation," IEEE Trans. on Human Factors in Electronics, Vol. HFE-6, No. 1, pp. 5-15 (March 1967).

10b7

8. D. C. Engelbart, W. K. English, and J. F. Rulifson, "Study For The Development of Human Intellect Augmentation Techniques," Interim Progress Report, Contract NAS1-5904, SRI Project 5890, Stanford Research Institute, Menlo Park, California (March 1967).

10b8

9. J. B. Hopper and L. P. Deutsch, "COPE: An Assembler and On-Line-CRT Debugging System for the CDC 3100," Technical Report 1, Contract NAS 1-5904, SRI Project 5890, Stanford Research Institute, Menlo Park, California (March 1968).

10b9

10. R. E. Hay and J. F. Rulifson, "MULY40: A Machine-Oriented ALGOL-Like Language for the SDS 940," Technical Report 2, Contract NAS 1-5904, SRI Project 5890, Stanford Research Institute, Menlo Park, California (April 1968).

10b10

11. D. C. Engelbart, W. K. English, and J. F. Rulifson, "Development of a Multidisplay, Time-shared Computer Facility and Computer-Augmented Management System Research," Final Report, Contract AF 30(602)4103, SRI Project 5919, Stanford Research Institute, Menlo Park, California (April 1968), AD 843 577.

10b11

12. D. C. Engelbart, "Human Intellect Augmentation Techniques," Final Report, Contract NAS 1-5904, SRI Project 5890, Stanford Research Institute, Menlo Park, California (July 1968), CRSTI Order No. N68-16140.

10b12

13. D. C. Engelbart, W. K. English, and D. A. Evans, "Study for the Development of Computer-Augmented Management Techniques," Quarterly Progress Report 1, Contract F30602-68-C-0280, SRI Project 7101, Stanford Research Institute, Menlo Park, California (October 1968).

10b13

14. D. C. Engelbart and W. K. English, "A Research Center for Augmenting Human Intellect," in AFIPS Proceedings, Vol. 33, Part One, 1968 Fall Joint Computer Conference, pp. 395-410 (Thompson Book Co., Washington, D.C., 1968).

10b14

Network Information Center  
and Computer Augmented Team Interaction

15. D. C. Engelbart and Staff of the Augmented Human Intellect Research Center, "Study for the Development of Human Intellect Augmentation Techniques," Semiannual Technical Letter Report 1, Contract NAS 1-7897, SRI Project 7079, Stanford Research Institute, Menlo Park, California (February 1969). 10015

16. D. C. Engelbart, W. K. English, and D. A. Evans, "Study for the Development of Computer Augmented Management Techniques," Interim Technical Report RADC-TR-69-48, Contract F30602-68-C-0286, SRI Project 7101, Stanford Research Institute, Menlo Park, California (March 1969), A 055 579. 10016

17. D. C. Engelbart and Staff of the Augmented Human Intellect Research Center, "Study for the Development of Human Intellect Augmentation Techniques," Semiannual Technical Letter Report 2, Contract NAS 1-7897, SRI Project 7079, Stanford Research Institute, Menlo Park, California (August 1969). 10017

18. D. C. Engelbart and Staff of the Augmented Human Intellect Research Center, "Augmentation Systems and Information Science," SRI Project 5890, sound film of presentation at ASIS Annual Meeting, October 1, 1969. 3 reels, 1 hour 34 min. 10018

19. D. C. Engelbart and Staff of Augmentation Research Center, "Computer-Augmented Management-System Research and Development of Augmentation Facility," RADC-TR-82, April 1970, Final Report of Contract F30602-68-C-0286, SRI Project 7101, Stanford Research Institute, Menlo Park, California. 10019

20. D. C. Engelbart, "Intellectual Implications of Multi-Access Computer Networks," paper presented at the Interdisciplinary Conference on Multiple-Access Computer Networks, Austin, Texas, April 20-22, 1970. 10020

21. D. C. Engelbart and Staff of Augmentation Research Center, "Advanced Intellect-Augmentation Techniques," Final Report NASA Contract NAS1-7897, July 1970, SRI Project 7079, Stanford Research Institute, Menlo Park, California. 10021

Note: Reports with AL numbers are available from Defense Documentation Center, Building 5, Cameron Station, Alexandria, Virginia 22314. Reference Nos. 6 and 12 may be obtained from CFSTI, Sills Building, 5825 Port Royal Road,

Network Information Center  
and Computer Augmented Team Interaction

SRI-ARC 1 JULY 71 8277  
References

Springfield, Virginia 22151; cost \$3.00 per copy or 65  
cents for microfilm.

10b22

Network Information Center  
and Computer Augmented Team Interaction  
70

APPENDICES	11
A. APPENDIX A, I/O BOX	11a
1. I/O CONTROL SYSTEM	11a1
2. General	11a2

The I/O control box connects onto the PDP-10 I/O system and is used to interface control signals and interrupt signals between various external devices and the PDP-10.

### 3. CONN To Devices

The PDP-10 controls external devices through the execution of a CONO instruction with device code 420. 11a3a

The right half of the word has the following format.

18	21	32	33	35
---	---	---	---	---
: ignore :	12 bits	:	:	
---	---	---	---	---
	sub-device bits	order code	llab3a18	

by setting bits 21 through 32, the order code can be transmitted to any number up to 12 external devices.

bits 33 through 35 are decoded to generate one of eight commands that can be transmitted to the indicated devices. 11a3a2 11a3a3

order code 0 has been reserved to represent a  
reset command.

11a3a3a

In general only the first four order codes have been decoded in the hardware.

11a3a3b

when the "RESET" switch on the PDP-10 consol is pushed the order code 0 is transmitted to all 12 devices.

1183a4

Bit assignment within this field as well as order functions are defined below.

11a3a5

bit	device	order	code	function
32	Disc/Drum System	0	0	reset system
			1	reset drum
			2	reset disc
			3	start drum
			4	Go chain disc
			5	Go no-chain disc
31	Display System 1	0	6	Disconnect disc
			0	reset
			1	initiate
			2	pause
			3	restart
			0	reset
30	Display System 2	0	1	initiate
			2	pause
			3	restart
			0	reset
			1	initiate
			2	reset
29	I.D.C.	0	3	restart
			0	reset
			1	initiate
			0	reset
			1	initiate
			0	reset
28	Printer	0	1	initiate
			1	reset
			0	initiate
			1	reset
			0	timer
			1	receive
27	Network	0	2	send
			1	reset
			2	timer
			3	receive
			0	send
			1	reset
26	n.S.Data Set	0	0	initiate
			1	reset
			0	initiate
			1	reset
			0	initiate
			1	reset
25	unused			
24				
23				
22				
21				

#### 4. CONI FROM DEVICES

The PDP-10 can sample the state of various external devices through the execution of a `CONI` instruction with device code of `420`.

The right half of the word has the following format.

18 : ----- 35  
: : 10 bits :  
: -----  
: Device status lines : llamala

Complete flexibility is allowed in connecting any

Network Information Center  
and Computer Augmented Team Interaction  
74

status condition of any device to some particular bit  
within this field. 11a4a2

bit assignments within this field are defined below. 11a4a3

bit device and condition

35	Drum busy
34	Disc busy
33	Disc error
32	Display 1 busy
31	Display 1 error
30	Display 2 busy
29	Display 2 error
28	I.D.C. busy
27	I.D.C. error
26	Printer busy
25	Printer error
24	Network busy
23	Network error
22	H.S.D.S. busy
21	H.S.D.S. busy
20	
19	
18	

11a4a3a

5. Interrupt Handling 11a5

The PDP-10 controls both the interrupt level and the masking of those devices from which it seeks interrupts. Control is executed through several CUNI and CUNO instructions to the I/O control box. 11a5a

Flag register 11a5b

The flag register stores the bits which are trying to generate an interrupt to the PDP-10 system. 11a5b1

This register can be sampled by the execution of a CUNI instruction with a device code of 414. 11a5b2

Data will be presented with the following format. 11a5b2a

18	29	35
---	---	---
:	: ignore	:
---	---	---
	flags	

11a5b2a1

Bits 18 through 29 are set when an interrupt has been requested from the appropriate device. 11a502b

Devices are assigned to bit positions according to the following table. **11a5b2c**

BIT	DEVICE
29	Bryant Disc:abnormal interrupt
28	Bryant Disc:normal interrupt
27	Display System 1
26	Display System 2
25	I.D.C.
24	Printer
23	Network - input
22	Network - output
21	H.S.D.S.
20	
19	
18	XCORE failure
	11a5b2c1

This register can be modified by the PDP-10 through the execution of a `CUENO` instruction with a device code of `414`. 11a503

The right half of the instruction has the following format. 11a5b3a

18 29 30 31 32 33  
--- --- --- --- --- ---  
: : : : ignore :  
--- --- --- --- --- ---  
flags control 11a5b3&1

Bits 10 through 29 indicate the bits of the flag register to be effected. 11abbbdb

If bit 30 is set, then the indicated bits of the flag register are to be set to zero. llabojc

If bit 31 is set, then the indicated bits of the flag register are to be set to one. 11ab303d

If bit 32 is set, then all the bits of the flag register are to be set to zero. 11ab53e

Mask A register 11ab5c

This register contains a 12 bit mask and a 3 bit interrupt level register. An interrupt is generated on the appropriate priority interrupt channel when a one occurs both in the flag register and in the mask A register. 11ab5cl

The source of an interrupt due to mask A can be determined through the execution of a CONI instruction with a device code of 400. 11ab5cla

Data will be returned with the following format. 11ab5clal

18	29	35
-----	-----	-----
: 12 bits	: ignore	:
-----	-----	-----
MASK A and flags		11ab5cla2

Bits 18 through 29 will be returned as ones only if both a bit for mask A and the corresponding flag bit are set. 11ab5cla3

The mask A register can be modified through the execution of a CONW instruction with a device code of 400. 11ab5c2

The right half of the instruction has the following format. 11ab5c2a

18	29 30 31 32 33	35
-----	-----	-----
:	:	:
-----	-----	-----
MASK A		control priority 11ab5c2al

Bits 18 through 29 indicate the bits of Mask A to be affected. 11ab5c2b

If bit 30 is set, then the indicated bits of the mask are to be set to zero. 11ab5c2c

If bit 31 is set, then the indicated bits of the mask are to be set to one. 11a5c2d

If bit 32 is set, then the interrupt level register is to be updated with the contents of bits 33 through 35. 11a5c2e

This register can be sampled through the execution of a DATAI instruction with a device code of 400. 11a5c3

Data is returned in the following format. 11a5c3a

16	29	33	35
--- -	: ignore	:	:
--- -	mask A	priority	11a5c3a

Bits 16 through 29 indicate the state of mask A. 11a5c3b

Bits 33 through 35 indicate the interrupt level set for mask A. 11a5c3c

mask B register 11a5c4

This register contains a 12 bit mask and a 3 bit interrupt level register. An interrupt is generated on the appropriate priority interrupt channel when a one occurs both in the flag register and in the mask register. 11a5c4

The operation of this mask register is identical to that of the mask A register with the provision that the device code for the appropriate COUN, CUNI, and DATAI instructions is 404. 11a5c5

UNIVAC Drum 11a5c6

Interrupts for the UNIVAC drum are handled separately from the other devices to allow for a unique interrupt level for this device. 11a5c6

An interrupt is generated on the appropriate interrupt level if the drum flag is set. 11a5c7

The state of the drum flag bit can be sampled through the execution of a CUNI instruction with a device code of 410. 11a5c8

SRI-ARC 1 JULY '71 8277  
Appendix A, I/O Box

Data is returned with the following format, `label2al`

Bit 29 is returned as a one if the drum flag bit is set. label2a3

The Drum flag and priority interrupt level can be modified through the execution of a COUNO instruction with device code 410 11a5e3

The right half of the instruction has the following format 11a5e3a

18		30	31	32	33	34
---	-----	-----	-----	-----	-----	-----
:	ignore	:	:	:	:	:
---	-----	-----	-----	-----	-----	-----
		control	priority	label	label	label

bit 31 will set the Drum flag. 11a5e3c

If bit 32 is set, the the priority interrupt level will be set to the value contained in bits 33 through 35. **llage3d**

The drum interrupt level can be sampled through the execution of a DATA1 instruction with a device code of 410. labeled

Data is returned with the following format.

16 : ignore : priority 33 35 : liaison

bits 33 through 35 indicate the disc priority interrupt level. **llase4b**

SKI-ARC 1 JULY 71 8277  
Appendix A, I/O Box

The drum interrupt can be turned off by setting  
the priority level to zero. 11a5e4c

Network Information Center  
and Computer Augmented Team Interaction  
70

B. APPENDIX B, UNIVAC DRUM SYSTEM

11b

1. General

11b1

The subsystem described here consists of 4 high-speed UNIVAC drum units model FH-432, and a UNIVAC drum control unit model 5012, connected to a PDP-10 memory bus through a special Disc-Drum Channel Logic unit. 11b1a

The total storage available on the 4 drum units is 1,048,576 words with an average access time of 4.3 milliseconds and a transfer rate of 240,000 words/second. 11b1b

The Disc-Drum Channel Logic processes commands to the drum by reading a Unit Reference Cell (URC) in memory for instructions. In addition it allows the Bryant Disc controller to share access to memory through the same memory bus. 11b1c

In addition to acting as a drum controller/interface, the Disc-Drum Channel Logic also connects the Bryant Disc System with the PDP-10 memory. Memory access is multiplexed between the disc and drum a cycle at a time where the drum has high priority. 11b1c1

The Disc-Drum Channel Logic is connected to the PDP-10 memory through the high priority port of the DEC MA-10 memory modules. 11b1c2

The drum URC is a fixed, three-word block of computer core memory. 11b1d

URC	64	function word for drum
URC 1	65	word count and memory address
URC 2	66	status message

11b1d

2. CONO, CONI, and Interrupt Instructions

11b2

Three CONO instructions are defined for the disc subsystem. 11b2a

The CONO codes are (device code 420) 11b2a1

742200 000010	Reset Disc/Drum System
742200 000011	Reset Drum
742200 000013	Start Drum

11b2a1a

The CONO actions are: 11b2a2

SRI-AFC 1 JULY 71 0277  
Appendix B, UNIVAC DRUM System

Start Drum -- This CONO causes the controller to execute the command contained in the URC. 11b2a2a

Command processing consists of fetching the control words from memory, transmitting the function word to the drum, and managing the resulting data transfers between memory and drum. 11b2a2a1

A Start Drum CONO issued while the system is busy will be ignored. 11b2a2a2

Reset Drum -- This CONO immediately terminates any drum operation in process when the CONO is received, and returns the system to the disconnect state. 11b2a2b

Reset Disc/Drum System -- This CONO immediately terminates any disc or drum operation and return the entire disc/drum channel logic to the reset state. 11b2a2c

One CONI condition is sensed. 11b2b

The CONI device code is 420 11b2b1

742240 YYYYYYYY Sense input conditions 11b2b1a

Bit 35 -- This bit is set to a one if the drum system is busy 11b2b1b

Drum Interrupt 11b2c

An interrupt is generated on the appropriate interrupt level of the Drum Flag is set. 11b2c1

The Drum Flag and priority interrupt level can be modified through the execution of a CONO instruction with a device code of 410. 11b2c2

Bit 30 set will reset the Drum Flag 11b2c2a

Bit 31 set will set the Drum Flag. 11b2c2b

If bit 32 is set, the priority interrupt level will be set to the value contained in bits 33 through 35. 11b2c2c

A more complete description of the CON0, CON1 and interrupt capability for special hardware devices can be found in the I/O CONTROL BOX section of the appendix. 1102d

3. URC Processing 1103

During the command table processing sequence, the second word of the URC will be fetched first. 1103a

The second word of the URC has the following format. 1103a1

	URC+1	
0	17 18	35
:	:	:
	Word Count	Memory Address 1103a1a

Bits 0 - 17 A positive word count including the value zero. 1103a1b

Bits 18-35 are an 18 bit address indicating the first word in PDP-10 memory for the current transfer. If this address is to be extended to 20 bits for use with the B&N paging box, the two additional bits are to be found in the first word of the URC. 1103a1c

If either a zero word count or a memory parity error is detected while reading this word of the URC, the Status word will be written indicating such an error and the process terminated with no command sent to the drum. 1103a2

After reading the first word of the URC and finding a non-zero word count, the first word containing the drum command is read. 1103b

This word has the following format.

0	5	12	14	15	17
:	:	:	:	:	:
Fun. Code	Ignore	Ident	Drum	Unit	
18	24 25		35		
:	:		:		
Band No.	Angular Address		11b3b1b		

Bits 0 - 5 This is a function code to be sent to the drum controller. Only 5 codes are acceptable and all others will result in terminating with an appropriate error bit in the channel status report. The allowed functions are described below. 11b3b2

02	Continuous Write	
42	Read Normal	
41	Read Early	
43	Read Late	
63	Send Angular Address	11b3b2a

Codes 02 and 42 are normally used to write and read with the drum. 11b3b2b

Codes 41 and 43 are the same as the Continuous Read (42) function except that the drum read probes are shifted to read data pulses slightly earlier or later. These functions can be used to try to recover data following a parity error, or to aid online maintenance. 11b3b2c

Code 63 is used to instruct the UNIVAC controller to send a status word containing the current angular address of the drum specified by the function word. This is a special command in as much as the channel logic ignores the word count field. (this field must be non-zero however so that the Channel logic will read this word in the URC). 11b3b2d

The Angular report is based on the selected Drum Unit. The remaining bits of the drum address (10-35) will be ignored. 11b3b2d1

In most cases, the interrupt is returned within about 30 microseconds after the CONO is issued. If the "dead space" is under the read head when the function is in progress, up to 230 microseconds may elapse. 11b3b2a2

If this function word addresses an inoperable drum unit, the status word containing the Illegal Address (54) status code is returned. 11b3b2a3

The format of the Angular Position Report is described under the Status word. 11b3b2a4

Bits 12-14 This is an ident field which must be set to either all one's or all zero's. 11b3b3

Bits 15-35 These bits represent the drum address as interpreted by the UNIVAC Controller. 11b3b4

If the Channel Logic detects either an illegal command or a parity error the operation will be terminated with appropriate bits set in the Status word. 11b3b5

After processing the two words in the URC, the Channel will then proceed to transfer data until the word count becomes zero. At this point a Control Code of 33 is generated and sent to the UNIVAC Controller so as to conclude the current function and return the drum status to the Channel Logic. The drum status information is used by the Channel logic in updating the status word in the URC. 11b3c

#### 4. Status Report 11b4

Before setting the drum flag at the completion of a command, the third word in the URC is updated by the Drum Control Logic. 11b4a

This word will have the following format. 11b4a1

				URC+3
0	5	6	11 12	35
---	---	---	---	---
:	:	:		:
---	---	---	---	---
status	channel		drum information	
code	status			

11b4a1a

SRI-ARC 1 JULY 71 8277  
Appendix B, UNIVAC Drum System

Bits 0-5 The status code which is returned from the UNIVAC controller will have only those values described below. 1104a2

(00) - Channel Fault 11b4a2a

An error was detected by the Disc-Drum Controller such that no request was passed on to the UNIVAC system. 11b4a2a1

The error detected is indicated within the Channel Status portion of the Status word. 11b4a2a2

The contents of the 24 low-order bits of the status word are indeterminate and should be ignored. 1104a2a3

(14) - Fault 1104a2b

The Fault status code is used to inform the processor that a hardware malfunction has occurred in the subsystem. Conditions which can cause a Fault indication are: 11b4a2b1

More than one read-write head has been selected. 11b4a2b1a

Power to the drum units has been interrupted during the operation. 11b4a2b1b

Angular address circuits in an FH-432 drum unit are out of synchronization. 11b4a2b1c

The WRITE VOLTAGE switch in the control unit is OFF when any function was received. 11b4a2b1d

The contents of the 24 low-order bits of the status word are indeterminate and should be ignored. 11b4a2b2

This error code can result from any of the valid function codes used on this system. 11b4a2b3

(20) - Angular Address 11b4a2c

The Angular Address status code is sent to the processor in response to Send Angular Address function (22). For the FH-432 drum unit, the 11 low-order bits of the status word contain

the angular address present about 10 microseconds before the time the drum flag interrupt signal was turned on. 1104a2c1

(34) : End-of-file 1104a2d

The End-of-file status code is used to inform the processor that the next sequential address is outside the set of legitimate drum addresses of the particular subsystem, is on an inoperable drum, or is on logical drum unit 1 for a write function when a WRITE LOCKOUT switch is set and applied to drum unit 1. 1104a2a1

This status code is generated only through increment of the drum address during a function. 1104a2a2

A status word containing an End-of-file status code is generated in response to any of the valid function codes except send Angular address. 1104a2a3

The contents of the 24 low-order bits of the status word are indeterminate and should be ignored. 1104a2a4

(40) - Normal Completion 1104a2e

If a Normal Completion is generated at the end of a data transfer, then the previous function was completed without an error detected. 1104a2e1

The contents of the 24 low-order bits of the status word are indeterminate and should be ignored. 1104a2e2

(54) - Illegal Address 1104a2f

The Illegal Address status code is used to inform the processor that the drum address in the function word is invalid. 1104a2f1

An invalid address is defined as an address specified in any read or write function word which is not within the set of legitimate addresses for the subsystem or which is on an inoperable drum. 1104a2f2

An address specified in a write function word which is in the set of addresses locked out by a WRITE LOCKOUT switch is also designated as an invalid address. 11b4a2i3

If a function word specifies an invalid address, the function is not initiated, and no data is transferred to or from the drum. 11b4a2f4

The contents of the 24 low-order bits of the status word are indeterminate and should be ignored. 11b4a2i5

(6X) - Parity Error 11b4a2g

The Parity error status code is used to inform the processor that the control unit detected a parity error during a read operation. The 24 low-order bits of the status word contain the drum address of the word in which the error was detected. 11b4a2g1

If a data parity error is detected, the status word is made available to the processor, and the interrupt signal is turned on only after the processor has accepted all parity-correct data words read for input to the processor before the error was detected. The error word is not made available to the processor. 11b4a2g2

The following procedure is recommended in attempting to recover from a parity error condition. 11b4a2g3

Initiate a continuous Read (42) function and check whether the parity error persists. 11b4a2g3a

If the parity error is reported, initiate a Read Early (41) function. 11b4a2g3b

If the parity error persists, initiate a Read Late (43) function to check again for correct parity. 11b4a2g3c

If the parity error is the response received for each step of the recovery procedure, then the error must be considered a non-recoverable drum error. 11b4a2g3d

C. APPENDIX C, BRYANT DISC SYSTEM

llc

1. General

llcl

The subsystem described here consists of a Bryant Disc File Series 4000, mod A2A, and a control unit. The present 7-disc system is capable of storing approximately 23 million 36-bit words.

llcl

The disc Unit Reference Cell (URC) is a fixed three-word block of computer core memory.

llclb

URC	70	pointer to command table
URC+1	71	advance sector information
URC+2	72	error message

llclb

All words in the URC and the command table as used by the disc controller are 24-bit fields corresponding to bits 12 through 35 of the PDP-10 word format. Bits 0 through 11 will be ignored by the controller and returned as zeros when writing into core.

llc1c

Data transferred to or from the disc will be 36-bit words plus odd parity.

llc1d

2. CONO and CONI Instructions

llc2

Five CONO instructions are defined for the disc subsystem.

llc2a

The CONO codes are (device code 420)

'742200 000010	Reset Disc/drum system
'742200 000012	Reset Disc
'742200 000014	Go chain
'742200 000015	Go no-chain
'742200 000016	Disconnect

llc2a1

The CONO actions are:

llc2a2

Reset Disc/drum system -- This CONO immediately terminates any disc or drum operation which may be in process when the CONO is received, and returns the Disc/drum System to the disconnect state.

llc2a2a

The digit designated "x" can be any number from 1 through 7 to signify the portion or portions of the word containing the parity error. 11042264

**Status Code      Incorrect Parity**

61 24 through 35  
62 12 through 23  
63 12 through 23 and 24 through 35  
64 0 through 11  
65 0 through 11 and 24 through 35  
66 0 through 11 and 12 through 23  
67 All three 12-bit segments 11b4a2g4a

bits 6-11 This field is used by the Channel Logic to indicate any fault conditions that it may detect. The bits used and the corresponding errors are listed below. lib4a3

Bit 0 -- Bad End 1104a38

The UNIVAC drum controller indicates a not ready state, does not complete a command, or is not plugged into the Channel Logic. llb44jal

Bit 7 -- parity error 1104a3b

The Channel Logic detected a parity error when reading PDR-10 memory. lib4a301

Bit 8 -- Illegal Function 1104a3c

The first word in the BOC contained an illegal function code. 1104a3c1

Bit 4 - Drum non-exempt

The PUP-10 memory address accessed by the drum portion of the Disc-drum channel logic did not respond within 100 microseconds. This failure indicates either an illegal memory address or a malfunctioning memory unit. 1104a501

Bits 10 - 11 -- Not Used 1104age

These bits are currently not used and will always be returned as zeroes. 1104a,el

Go-Chain -- This CONO causes the controller to start command processing. l1c2a2b

Processing always starts with the command addressed by the URC when the CONO is executed. l1c2a2b1

If a disconnect request has previously been stored by a Disconnect CONO and the system is still busy (processing commands), a Go-Chain CONO cancels the disconnect request. l1c2a2b2

A Go-Chain CONO issued while the system is busy and no disconnect request is stored results in a command error. l1c2a2b3

Go-No chain -- This CONO causes the controller to process the single command table entry pointed to by the URC. l1c2a2c

A Go-No Chain CONO received while the controller is processing commands results in a command error. l1c2a2c1

Reset -- This CONO immediately terminates any disc operation in process when the CONO is received, and returns the system to the disconnect state. l1c2a2c2

Disconnect -- This CONO causes the controller to disconnect at the next normal interrupt condition. l1c2a2d  
Two CONI conditions are sensed. l1c2b

The CONI device code is 420

742240 YYYYYYYY Sense input conditions

The conditions sensed are; l1c2b1  
l1c2b2

Bit 34 -- This bit is set to a one if the disc system is busy l1c2b2a

Bit 33 -- This bit is set to a one if any outstanding error conditions exists on the disc subsystem. Execution of this instruction does not reset any error conditions. l1c2b2b

The execution of a Go-Chain C0NO before the next normal interrupt condition is reached cancels the disconnect request. llc2e

### 3. Command-Table Processing

llc3

After either Go C0NO the system begins processing commands with the command addressed by the URC. llc3a

The URC always points to the current command being processed. llc3al

In a Go-Chain or Go-No Chain operation, after the successful completion of the command, the URC is updated (incremented by 3) to point to the first word of the next command. llc3ala

There are three types of commands in the command table. llc3b

Data Transfer Command -- This command consists of three command words in contiguous memory locations. llc3b1

The first word contains the disc address. It consists of concatenated binary address fields. Not all combinations in certain address fields are used; the unused combinations form invalid addresses. The address word has the following format:

1415	22 24	31	35		
-----	-----	-----	-----		
:1 0: :	: : :	: :	:		
-----	-----	-----	-----		
I	Track	Zone	Head	sector	llc3b1

Interrupt bit -- If Bit 14 is a 1, a normal interrupt is given after successful completion of the command. llc3b1b

Track Address field (8 bits) -- This field is used to select one of 256 head array positions. All bit combinations in this field are valid.

llc3b1b

Zone Address Field (2 bits) -- This field is used to select one of the three disc frequency zones as follows:

00	Zone 0	
01	Zone 1	
10	Zone 2	
11	Invalid	llc3b1b2

Head Address Field (7 bits) -- This field is used to select one of the 26 data heads in the specified zone. llc3b1b3

Heads are numbered 0 to 25, and are arranged two per physical surface per zone llc3b1b3a

The valid addresses for the 6 disc system are 0000000 through 0011001. llc3b1b3b

Sector Address Field (4 bits) -- This field is used to select the proper sector on a track. llc3b1b4

The valid combinations for this field depend on the zone selected. Sectors are numbered zero to k, where k is one less than the number of sectors in the zone. The following combinations for each zone are valid. llc3b1b4a

Zone	Address Field	Sectors
1	0000-0001	2
2	0000-0100	5
3	0000-0110	7

llc3b1b4al

The second word contains the class and word count. Its format is as follows:

12	10	24	35
-----	-----	-----	-----
:	:	:	:
-----	-----	-----	-----
	Class	Count	llc3b1b4c

Class Field contains the Direction-of-Transfer Bit (Read/Write) and information on headers. It is subdivided as follows: llc3b1b4d

12	13	14	18
-----			
:	:	:	:
-----			
Head	I/O	Class	llc3b1b4al

head -- If this bit is a 1, header fields are written with the record. llc3b1b4e

I/O -- These bits determine the direction of transfer and the use of the class field as follows: llc3b1b4f

00	Read - No compare with class	
01	Read - Compare with class	
10	Write record and class field	
11	Write if class compares equal	llc3b1b4f1

Class -- This 4-bit field appears in each record defining a class to which the record belongs. If class comparison is called for and fails, an error interrupt is given. llc3b1b4g

Count Field -- This field defines the number of 36-bit words to be transferred. llc3b1b45

The maximum word count is 2040. Exceeding this count in the command word results in an illegal word count error. llc3b1b4a

If the field is zero the command serves to position the head array only. (Headers may be written with a word count of zero). llc3b1b4b

The third word contains the core memory address at which the transfer is to begin. The word format is:

12	10	35
-----		
:	:	:
-----		
	Core Address	llc3b1c1

Core Address -- This field contains the absolute core address at which the information transfer is to begin. 11c3b0d

Branch Command -- This command causes the next command word to be taken from the core location given in the branch command word rather than in sequence in the command table. The core address is absolute and no remapping takes place. The word format is: 11c3b2

12	14	19	35
-----			
:0	1:	:	:
-----			
I		Core Address	11c3b2a

If the interrupt bit is set a normal interrupt will be generated after the command is executed. 11c3b2b

Note: After a branch command the URC is written with the entire contents of the branch command word. 11c3b2c

Disconnect Command Word -- This word causes the disc controller to disconnect. The word format is: 11c3b3

12	14	35
-----		
:0	1:	:
-----		
I		11c3b3a

If the interrupt bit is set a normal interrupt will be generated after the command is executed. 11c3b4

#### 4. Disc File Formats 11c4

Disc Format: Each of the twelve data surfaces is divided into three zones, with a pair of heads for each zone. Each of the three zones has a separate clock frequency and bit density optimized for the zone. 11c4a

Zone Format: A zone is divided into 512 tracks, corresponding to each of two heads at 256 positions of the head array. 11c4b

SRJ-ARC 1 JULY '71 0277  
Appendix C, Bryant Disc System

Track Format: A track is divided into sectors by  
prerecorded sector pulses. The number of sectors per  
track is a function of the zone. llc4c

Zone 0	2 sectors/track	Inner Zone
Zone 1	5 sectors/track	Middle Zone
Zone 2	7 sectors/track	Outer Zone

llc4cl

Sector Format: There is one fixed-length record per  
sector with a data field of 256 30-bit words.  
Associated with each record is a header field used to  
identify the record and ensure that head and zone  
selection are correct before writing or reading a  
record, and a class field grants access to records by  
class. llc4d

In all subfields of the sector a preamble and  
postamble ensure reliable reading of the first and  
last bits of the subfield. llc4d1

These bits are all "ones," generated by the  
controller and never transferred to the computer.  
The overall format of the sector is llc4d2

74 bits	37 bits	9546 bits
-----	-----	-----
:	:	:
-----	-----	-----
Header Field	Class Field	Data Field

llc4d2a

The header field consists of two header words  
generated by the control unit and is not transferred  
to the Central Processor. llc4a3

These words are only written when special key  
switches (one for each header word) are on and a 1  
appears in the 0 bit of the class and count word.  
Header word 1 llc4d3a

8 bits	8 bits	5 bits	1	3 bits
-----	-----	-----	-----	-----
: Preamble	: Track Address:	Zeroes	P	: Postamble:
-----	-----	-----	-----	-----

llc4d3b1

This word is written by the disc controller and is  
used for track verification.

llc4a3c

Header Word 2

8 bits 2 7 bits 4 bits 1 3 bits  
-----  
: Preamble : Z : surface :sector : P :postamble:  
-----

llc4d3a1

Zone Subfield (2 bits) -- These two bits  
correspond to the zone address and are used to  
insure proper selection of the zone. llc4d3a2

Head Subfield (7 bits) -- These seven bits are  
used to ensure correct selection of the head.  
Heads are arranged two per physical surface per  
zone. llc4d3a3

Sector Subfield (4 bits) -- This subfield is  
used to identify the sector or record and is  
unique on each track. llc4d3a4

Parity Subfield (1 bit) -- odd parity is  
generated for each header word and is checked  
whenever the header is read. llc4d3a5

Class Field Format -- The format of the class  
field is:

8 bits 4 bits 9 bits 1 3 bits  
-----  
: Preamble : Class : Zeros : P : Postamble:  
-----

Class Subfield -- This is a 4-bit field defining the class to  
which a record belongs. Normally the class  
field is read and compared with that appearing  
in the command word; if they are equal the  
operation proceeds. llc4d3e2

Parity Subfield (1 bit) -- Odd parity llc4d3e3

Data Field Format

0 bits 9472 bits 9 bits 3 bits  
-----  
: Preamble : Data : Check Bits : Postamble:  
----- llc4d3f1

Data Subfield (9472 bits) -- This subfield  
consists of 256 36-bit machine words. An odd  
parity bit is inserted every 36 bits by the  
control unit. It is transferred in its  
entirety on a read operation with odd parity  
generated for each word. If less than 256  
words are transferred on a write, the control  
unit generates the necessary zeros to fill out  
the data subfield. llc4d3f2

Check Subfield (9 bits) -- This subfield is  
used for error checking over the data record.  
It is generated by the control unit on a read  
or write operation and is never transferred to  
the central processor. llc4d3f3

Gap Format -- A gap of 111 bit times is allowed  
between each alterable segment of the sector  
format and the next. This allows sufficient time  
for the recovery of the read amplifiers after  
writing a segment of the sector field. llc4d3g

5. Clocking llc5

Clock tracks are prerecorded on a separate disc with its  
own set of heads which do not move. llc5a

Each zone has a separate heads for write clock and  
sector/index pulse. llc5a1

When the system is busy, the advance sector word is  
updated by the controller to indicate the next  
available sector in each zone. This word has the  
following format.

12 1516 23 27 31 35  
-----  
: : : : : : :  
-----  
TV Track Zone 3 Zone 2 Zone 1 llc5a24

"TV" is the track verification bit. When this bit is a 1 the heads have settled on the addressed track. llc5a2b

The "track" code indicates the head array position if TV is 1 and head array destination if TV is 0. llc5a2c

The advance sector information as described here has been turned off in the hardware due to difficulties in this portion of the controller. llc5a2d

#### 6. Error Conditions llc6

Whenever an abnormal condition is detected by the controller the following actions occur: llc6a

Any data transfer operation in process is terminated. llc6a1

A disc read operation is terminated immediately on detection of the error. llc6a1a

On a disc write operation the remainder of the current sector is filled with zeros and the operation is terminated. llc6a1b

Bits indicating the error conditions are written in the disc error word. llc6a2

An abnormal interrupt is generated. llc6a3

The controller goes to the disconnect state. llc6a4

The disc error word contains a 1 for every abnormal condition that has occurred. At least one bit will always be set and more than one can be set. llc6b

The format of this word is

bit

24	Illegal	
26	Control Unit error	
27	Class Not Equal	
28	Not Ready	
29	Angular Position Error	
30	Head Position Error	
31	Invalid Address	
32	Command Error	
33	Data Transfer Error	
34	Check Field Error	
35	Word Parity Error	llc6b1

Data and Command Errors llc6b2

Word Parity Error (Bit 35) -- This condition is set whenever the parity is incorrect on a 24-bit sequence in the data field of a record during a read operation. llc6b2a

Check Field Error (bit 34) -- This bit is set whenever the check bits at the end of the record indicate that an error has been made in reading the record. llc6b2b

Data Transfer Error (bit 33) -- This bit is set when data being transferred from the Central Processing Unit to the Control Unit has incorrect parity. llc6b2c

Command Error (bit 32) -- This bit is set for the following conditions: llc6b2a

    Incorrect parity for a command word transferred from the computer. llc6b2d1

    Invalid command code. llc6b2d2

    A Go-No Chain CONU received while busy. llc6b2d3

Go-Chain CONO receive while busy and no  
disconnect request waiting. l1c6b2d4

Addressing and Positioning Errors l1c6c

Invalid Address (bit 31) -- This bit is set when the  
disc address specified in a transfer command is  
invalid, or a data transfer exceeds the cylinder. l1c6c1

A cylinder consists of all tracks on all surfaces  
that can be accessed from a single head position.  
l1c6c1a

Head Position Error (bit 30) -- This bit is set if  
the head array is not correctly positioned as  
determined by failure to get track verification after  
7 revolutions or incorrect track address in header  
word 1. l1c6c2

Angular Position Error (bit 29) -- This bit is set  
when the angular position specified in the address  
does not match that read from header word 2, or if a  
parity error is detected in header word 2. l1c6c3

Illegal Word Count (bit 24) -- This bit is set when  
the word count in a data transfer command exceeds  
2048. l1c6c4

Miscellaneous Errors l1c6d

Not Ready (bit 28) -- This bit is set if the control  
unit receives an information transfer command and the  
disc is not ready. l1c6d1

Class Compare Not Equal (bit 27) -- This bit is set  
if a class compare is requested and the record has a  
different class from the Information Transfer  
Command. l1c6d2

Control Unit Error (bit 26) -- This bit is set when  
timing or sequencing errors in the control unit  
prevent completion of the operation. l1c6d3

l1c6d4